НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики

Кафедра програмного забезпечення комп'ютерних систем

«До захисту допущено»
Науковий керівник кафедри
_____ Іван ДИЧКА
«___» _____ 2020 р.

Дипломний проєкт

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Інженерія програмного забезпечення комп'ютерних та інформаційно-пошукових систем»

спеціальності 121 Інженерія програмного забезпечення

на тему: «Програмний застосунок для аналізу емоційного забарвлення зображень людини»

Виконав:
студент IV курсу, групи КП-61
Садрицький Сергій Володимирович _____

Керівник:
Доцент кафедри ПЗКС, к.т.н., доцент,
Сулема Євгенія Станіславівна _____

Консультант з нормоконтролю:
Доцент кафедри ПЗКС, к.т.н.,
Онай Микола Володимирович _____

Рецензент:
Доцент кафедри ПМА, к.т.н, доцент,
Сирота Сергій Вікторович _____

Засвідчую, що у цьому дипломному проєкті немає запозичень з праць інших авторів без відповідних посилань.

Студент _____

Київ – 2020 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет прикладної математики
Кафедра програмного забезпечення комп'ютерних систем

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 121 «Інженерія програмного забезпечення»

Освітньо-професійна програма «Інженерія програмного забезпечення комп'ютерних та інформаційно-пошукових систем»

«ЗАТВЕРДЖУЮ»
Науковий керівник кафедри
_____ Іван ДИЧКА
«___»_____2019 р.

ЗАВДАННЯ

на дипломний проєкт студенту

Садрицькому Сергію Володимировичу

1. Тема проєкту «Програмний застосунок для аналізу емоційного забарвлення зображень людини», керівник проекту Сулема Євгенія Станіславівна, к.т.н., доцент, затверджені наказом по університету від «25» травня 2020 р. № 1181- с

2. Термін подання студентом проєкту «15» червня 2020 р.

3. Вихідні дані до проєкту: див. Технічне завдання.

4. Зміст пояснювальної записки:
   - аналіз існуючих рішень;
   - розроблення програмного застосунку;
   - опис розроблених алгоритмів;
   - аналіз розробленого застосунку.

5. Перелік обов'язкового графічного матеріалу:
   - алгоритм визначення емоційного забарвлення (креслення);
   - взаємодія графічних елементів (креслення);
   - вікна програмного застосунку (плакат);
   - схема архітектури розробленого рішення (плакат).

6. Консультанти розділів проєкту

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Нормоконтроль	Онай М.В., доцент		

7. Дата видачі завдання «31» жовтня 2019 р.

Календарний план

№ з/п	Назва етапів виконання дипломного проєкту	Термін виконання етапів проєкту	Примітка
1.	Вивчення літератури за тематикою проєкту	17.11.2020	
2.	Розроблення та узгодження технічного завдання	7.12.2020	
3.	Розроблення структури застосунку	22.12.2020	
4.	Розроблення дизайну вікон та графічних елементів	14.01.2020	
5.	Програмна реалізація клієнтської частини застосунку	18.02.2020	
6.	Програмна реалізація серверної частини застосунку	7.03.2020	
7.	Програмна реалізація алгоритму визначення емоційного забарвлення	18.03.2020	
8.	Тестування застосунку	13.04.2020	
9.	Підготовка матеріалів текстової частини проєкту	11.05.2020	
10.	Підготовка графічної частини дипломного проєкту	24.05.2020	
11.	Оформлення технічної документації дипломного проєкту	30.05.2020	

Студент Сергій САДРИЦЬКИЙ

Керівник проєкту Євгенія СУЛЕМА

АНОТАЦІЯ

Даний дипломний проєкт присвячений розробленню програмного застосунку для розпізнавання і аналізу емоційного забарвлення зображень людини в режимі реального часу.

При розробці даного ПЗ було проведено порівняльний аналіз ряду існуючих альтернативних рішень для визначення емоційного стану користувача мережі, призначених для використання у рекламній та інших сферах, визначено сильні й слабкі сторони цих застосунків. Також було проаналізовано ряд засобів для веб-розробки, обґрунтовано вибір бібліотек та фреймворків для реалізації функціоналу у клієнтській та серверній частинах, обґрунтовано вибір бібліотеки для реалізації функції розпізнавання та аналізу зображень користувача в режимі реального часу, визначено систему керування базами даних, призначену для зберігання емоційних даних користувачів, що переглядають вміст веб-сторінок.

Даний застосунок розроблено у вигляді веб-додатку, що дозволяє рекламодавцям у ролі адміністратора мати доступ до даних про ставлення користувачів до наданої на сайтах рекламної інформації на основі емоційного аналізу методами штучного інтелекту. Ці дані рекламодавці можуть використовувати як основу для проведення аналітичних досліджень і визначення ефективних рекламних пропозицій у майбутньому.

ABSTRACT

This diploma project is dedicated to the development of a web-application for recognizing and analyzing the emotional colour of human images in real-time.

During the development of this project, a comparative review of some existing alternative solutions, which are designed for the use in advertising and other areas for the analysis of an network user's emotional state, was carried out. As a result, the strengths and weaknesses of these applications were identified. Also, several tools for web development was reviewed to substantiate the choice of suitable libraries and frameworks for implementing functionality in client and server parts. The additional analysis was conducted to make the choice of a library for implementing image recognition and analysis of user images in real-time, as well as a database management system intended to store emotional data of users viewing the content of web pages.

This application is designed as a web application that allows advertisers as administrators to access data about users' attitudes to advertising information provided on sites based on emotional analysis using artificial intelligence. These data can be used by advertisers as a basis for conducting analytical research and determining effective advertising offers in the future.

ДП.045440-01-90 Програмний застосунок для аналізу емоційного забарвлення зображень людини. Відомість проєкту

Позначення	Найменування	Кіл-ть	Примітка
	Документація проєкту		
ДП.045440-02-91	Програмний застосунок	5	
	для аналізу емоційного		
	забарвлення зображень		
	людини. Технічне		
	завдання		
ДП.045440-03-81	Програмний застосунок	50	
	для аналізу емоційного		
	забарвлення зображень		
	людини. Пояснювальна		
	записка		
ДП.045440-04-51	Програмний застосунок	4	
	для аналізу емоційного		
	забарвлення зображень		
	людини. Програма та		
	методика тестування		
ДП.045440-05-34	Програмний застосунок	6	
	для аналізу емоційного		
	забарвлення зображень		
	людини. Керівництво		
	користувача		
ДП.045440-06-99	Програмний застосунок	1	
	для аналізу емоційного		
	забарвлення зображень		
	людини. Схема взаємодії		
	графічних елементів		

Позначення	Найменування	Кіл-ть	Примітка
ДП.045440-07-99	Програмний застосунок	1	
	для аналізу емоційного		
	забарвлення зображень		
	людини. Визначення		
	емоційного забарвлення		
	користувача. Схема		
	алгоритму		
ДП.045440-08-98	Програмний застосунок	1	
	для аналізу емоційного		
	забарвлення зображень		
	людини. Компакт-диск		

Факультет прикладної математики

Кафедра програмного забезпечення комп'ютерних систем

«ЗАТВЕРДЖЕНО»

Науковий керівник кафедри

_____ Іван ДИЧКА

«___» _____ 2019 р.

ПРОГРАМНИЙ ЗАСТОСУНОК ДЛЯ АНАЛІЗУ ЕМОЦІЙНОГО ЗАБАРВЛЕННЯ ЗОБРАЖЕНЬ ЛЮДИНИ

Технічне завдання

ДП.045440-02-91

«ПОГОДЖЕНО»

Керівник проєкту:

_____ Євгенія СУЛЕМА

Нормоконтроль:

_____ Микола ОНАЙ

Виконавець:

_____ Сергій САДРИЦЬКИЙ

2019

ЗМІСТ

1. НАЙМЕНУВАННЯ ТА ГАЛУЗЬ ЗАСТОСУВАННЯ

Назва розробки: Програмний застосунок для аналізу емоційного забарвлення зображень людини.

Галузь застосування: інформаційні технології.

2. ПІДСТАВА ДЛЯ РОЗРОБЛЕННЯ

Підставою для розроблення є завдання на дипломне проектування, затверджене кафедрою програмного забезпечення комп'ютерних систем Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського» (КПІ ім. Ігоря Сікорського).

3. ПРИЗНАЧЕННЯ РОЗРОБКИ

Розробка призначена для використання в якості інформаційного забезпечення в сфері маркетингових досліджень з метою надання рекламодавцеві можливості аналізувати емоційну реакцію користувачів сайту на надані рекламні оголошення.

4. ВИМОГИ ДО ПРОГРАМНОГО ПРОДУКТУ

Програмний застосунок для аналізу емоційного забарвлення зображень людини повинен забезпечувати такі основні функції:

1) можливість аналізувати відеопотік у реальному часі;
2) можливість виявляти риси обличчя та розпізнавати емоції;
3) можливість збереження даних користувачів у базі даних;
4) можливість подавати аналітичну інформацію графічно;
5) можливість авторизації адміністратора для перегляду аналітики;

Розробку виконати за допомогою мови програмування JavaScript з використанням фреймворків NodeJS на стороні серверу та ReactJS на стороні клієнта.

Додаткові вимоги:

1) використання мінімалістичної колірної гами;

2) дизайн сторінок з використанням адаптивного підходу;

3) наявність логотипів на сторінках застосунку;

5. ВИМОГИ ДО ПРОЕКТНОЇ ДОКУМЕНТАЦІЇ

У процесі виконання проекту повинна бути розроблена наступна документація:

1) пояснювальна записка;

2) програма та методика тестування;

3) керівництво користувача;

4) креслення:

- «Алгоритм визначення емоційного забарвлення»;
- «Алгоритм взаємодії графічних елементів».

6. ЕТАПИ ПРОЄКТУВАННЯ

Вивчення літератури за тематикою роботи……..17.11.2019

Розроблення та узгодження технічного завдання…….....7.12.2019

Розроблення структури застосунку…….....22.12.2019

Розроблення дизайну вікон та графічних елементів…..……..14.01.2020

Програмна реалізація клієнтської частини…..……....18.02.2020

Програмна реалізація серверної частини…….…….....7.03.2020

Реалізація алгоритму визначення емоційного забарвлення…18.03.2020

Тестування застосунку…….……...…13.04.2020

Підготовка матеріалів текстової частини проєкту……..…11.05.2020

Підготовка графічної частини проєкту……..……...24.05.2020

Оформлення технічної документації проєкту……..…30.05.2020

7. ПОРЯДОК ТЕСТУВАННЯ РОЗРОБКИ

Тестування розробленого програмного продукту виконується відповідно до “Програми та методики тестування”.

Факультет прикладної математики

Кафедра програмного забезпечення комп'ютерних систем

«ЗАТВЕРДЖЕНО»

Науковий керівник кафедри

_____Іван ДИЧКА

«__» _____2020 р.

ПРОГРАМНИЙ ЗАСТОСУНОК ДЛЯ АНАЛІЗУ ЕМОЦІЙНОГО ЗАБАРВЛЕННЯ ЗОБРАЖЕНЬ ЛЮДИНИ

Пояснювальна записка

ДП.045440-03-81

«ПОГОДЖЕНО»

Керівник проєкту:

_____Євгенія СУЛЕМА

Нормоконтроль:

_____Микола ОНАЙ

Виконавець:

_____Сергій САДРИЦЬКИЙ

2020

ЗМІСТ

СПИСОК ТЕРМІНІВ, СКОРОЧЕНЬ ТА ПОЗНАЧЕНЬ

ПЗ − програмне забезпечення.

БД – база даних.

СКБД (Система керування базами даних) – набір лінгвістичних та програмних засобів спеціального чи загального значення, що забезпечують керування створенням та використанням баз даних [1].

UI (англ. User Interface) – графічний інтерфейс користувача, засіб зручної взаємодії користувача з інформаційною системою [2].

MVC (модель-представлення-контролер, англ. Model-View-Controller) – архітектурний шаблон побудови програмного забезпечення [3].

HTTP (англ. Hyper Text Transfer Protocol) – протокол передачі даних, що використовується і компютерних мережах. Передбачає клієнт-серверну взаємодію, при якій на кліет(часто браузер) передаються веб-сторінки або інші файли, з ними пов’язані [4].

JSON (JavaScript Object Notation) – текстовий формат обміну інформацією, що заснований на JavaScript нотації та призначений для описання обєктів та інших структур даних. Формат JSON представляє дані у вигляді «ключ-значення» [5].

API (Application Programming Interface) – прикладний програмний інтерфейс додатку, набір програмних засобів, таких як класи, процедури, функції, з допомогою яких одна комп’ютерна програма може взаємодіяти з іншою [6].

Computer Vision (з англ. *компютерний зір*) – засоби комп’ютерного зору, технології створення штучних систем, які оримують дані у вигляді зображень, що можуть бути представлені у вигляді багатьох форм, таких як

відеопотоки, тривимірні зображення, зображення з кількох камер та інші. Дані системи призначені для виявлення, стеження і визначення об’єктів [7].

Machine learning (з англ. *машинне навчання*) – галузь штучного інтелекту, що використовує його напрацювання для впровадження систем, здатних автоматично вдосконалюватись в процесі навчання без потреби у безпосередній передачі інструкцій [8].

AI (англ. Artificial Intelligence) – галузь комп’ютерних наук, що спеціалізується на розробці застосунків, які використовують програмні засоби для використання можливостей штучного інтелекту, таких як нейронні мережі, методи глибинного навчання та інші [9].

МЕТА ДИПЛОМНОГО ПРОЄКТУ

Метою даного дипломного проєкту є розробка програмного застосунку для аналізу емоційного забарвлення зображень людини.

Відповідно до мети дипломного проєкту було висунуто наступні завдання:

- з’ясування актуальності проблем цільової предметної області та визначення задач, що має на меті вирішити програмний застосунок; аналіз існуючих аналогів для рішення задачі емоційного аналізу зображень, визначення їх переваг та недоліків та на основі цього обґрунтування вимог до програмного застосунку;
- розгляд і порівняння засобів розробки програмного застосунку, таких як мови програмування, бібліотеки, фреймворки та системи керування базами даних;
- опис і графічне відображення архітектури, функціональних вимог, структури зберігання даних та особливостей реалізації додатку;
- контроль дотримання вимог до якості програмного застосунку; Виконання й опис процесу тестування застосунку; аналіз виконаної роботи на відповідність вимогам.

ВСТУП

Основна мета створення дипломного проєкту – розробка програмного застосунку для розпізнавання емоційного забарвлення зображень, який у дозволяє аналітикам з продажу на основі емоційного аналізу зображень обличчя споживача відстежити його реальне ставлення до наданих рекламних оголошень чи товарів.

Як відомо, в наш час стрімкого розвитку інформаційних технологій, зокрема, розростання об’ємів реклами в мережі Інтернет, користувачі все більше звертають увагу на засмічення простору нерелевантним рекламним контентом. Цілком природньо, знижується інтерес до інтернет-ресурсів, які заробляють рекламою(інтернет-магазини, блоги тощо) або ретранслюють її. Відповідно, існування і розвиток подібних рішень як ніколи актуальні як для потенційних користувачів, так і для тих, хто відповідальний за наповнення сайтів, адже репутація в медіа просторі ціниться високо.

Програмний застосунок має на меті стати інструментом рекламодавців для впровадження більш якісних і при цьому менш ресурсовитратних рекламних кампаній, адже, знаючи заздалегідь інтереси своїх потенційних клієнтів, можна бути впевненим, що це допоможе користувачам інтернет-ресурсів отримувати лише доцільні й потрібні цільові пропозиції.

Програмне забезпечення розроблене в даному дипломному проєкті надає наступні можливості:

- розпізнавання емоційного стану переглядача веб-сайту;
- збір масиву емоційних даних користувачів сайту;
- подання зібраних даних у зручному для обробки вигляді;
- представлення статистичних даних по сайту у зручному для перегляду форматі з використанням графіків, таблиць або діаграм;

- можливість вбудувати застосунок на довільний веб-сайт для перевірки його дії;
- зручний мінімалістичний UI.

Все вищезгадане дозволяє потенційним користувачам отримати зручний, а головне, інформативний програмний застосунок для збору й аналізу емоційних даних глядачів, який не потребує додаткових навичок програмування чи фінансових вкладень для користування.

1. ОГЛЯД ІСНУЮЧИХ РІШЕНЬ ТА ОБҐРУНТУВАННЯ ТЕМИ ДИПЛОМНОГО ПРОЄКТУ

1.1. Загальний опис проблеми емоційного аналізу зображень

Сучасний розвиток технологій не обходиться без розробок у сфері розпізнавання емоцій. Ринок таких систем активно зростає. За оцінками ряду експертів, до 2024 року спостерігатиметься середньорічний приріст на 21,4% і досягнення рівня 56,1 млрд. доларів [10].

Такі показники цілком виправдані, оскільки вже зараз програмне забезпечення для розпізнавання емоцій дозволяє визначати стан у довільний момент часу за допомогою веб-камери або спеціалізованого устаткування, паралельно аналізуючи поведінкові шаблони, фізіологічні параметри та зміну настрою користувача. Часто подібні застосунки дістають широке застосування у медицині, сфері безпеки, рідше - при слідчих процесах тощо.

Системи, що зчитують, транслюють та розпізнають дані емоційної природи, можна класифікувати в групах за типом визначення реакцій за фізіологічними показниками:

- за мімікою;
- за мовою тіла та рухами;
- за голосом.

Для прикладу, пристрій MindWave [11] виявляє емоції за фізіологічними даними, використовуючи датчики мозкової активності. Він фіксує ступінь концентрації, розслаблення або тривожності людини, оцінюючи їх за шкалою від 1 до 100. Інший приклад – сервіс FaceReader [12], що здатен інтерпретувати мікроекспресію обличчя, з високою точністю розподіляючи її типи на сім основних категорій: радість, сум, гнів, здивування, страх, відраза та нейтральне забарвлення.

Емоційний аналіз зображень – доволі складний процес, що включає кілька етапів збору, обробки даних та представлення зібраної аналітики користувачу. Принципи роботи подібних систем базуються на технологіях

комп’ютерного зору. Доволі популярним є метод класифікації за допомогою методів машинного навчання, зокрема нейронних мереж з використанням аналізу моделі обличчя по контрольних точках і врахуванням нюансів поверхні.

Даний курсовий проєкт буде охоплювати широке застосування, а для прикладу буде сфокусований на вирішенні проблеми формування доцільних рекламних пропозицій та запобігання розповсюдження нерелевантної цільової реклами у мережі. Останнім часом, левова доля найуспішніших рекламних кампаній зосередили свій вплив на емоціях споживача. Такі загальновідомі бренди як Apple, Nike, MasterCard та багато інших успішно користуються напрацюваннями на рубежі психології та IT [13]. В доповнення до вищевказаного, проєкт може знайти своє призначення й у інших областях, пов’язаних з аналізом людських емоцій, таких як визначення емоційного стану пацієнта в медицині чи кредитоспроможність платника у банківській сфері.

1.2. Аналіз існуючих рішень для даної задачі

Після аналізу деяких відомих аналогів ПЗ для вирішення даної проблеми було виявлено певні переваги та недоліки. Нижче наведено опис прикладів подібних застосунків.

Система розпізнавання емоцій людини EmoDetect

EmoDetect [14] – програмна розробка у сфері емоційного аналізу, що характеризується відносною точністю результату, широким діапазоном метрик та зручним графічним UI. Дана програмна система представлена у вигляді окремого додатку для аналізу відео або фото та програмного модуля API для вбудовування рощу сторонні веб-додатки.

Принцип роботи наступний: респондент дивиться на монітор, на якому демонструються стимули – рекламний ролик або упаковка продукту.

Веб-камера тим часом здійснює відеофіксацію і в реальному часі передає дані на обробку програмному модулю. На виході отримується

результат у формі таблиць та графіків. При порівнянні даних респондентів виявляються різні закономірності, що в подальшому впливають на рішення рекламодавця.

Переваги ПЗ:

- відносна швидкість обробки у реальному часі;
- використання за опорну точку нетрального вигляду обличчя;
- вивід статистики у вигляді графіків і таблиць.

Недоліки ПЗ:

- доступність ПЗ лише на території видавця – РФ;
- висока вартість користування;
- добре незадокументований API;
- нова система, не перевірена часом;
- для використання у сторонніх веб-додатках є вимога від користувача знань програмування.

Вигляд вікна програми EmoDetect наведено на рис. 1.

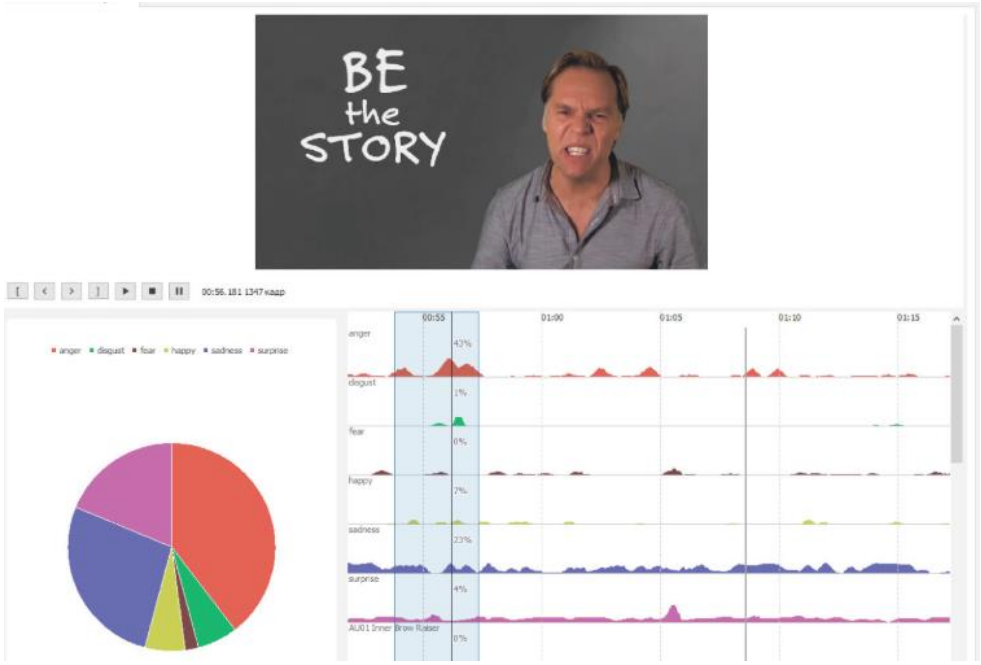


Рис. 1. Інтерфейс програми EmoDetect [14]

Застосунок nViso

Застосунок **nViso** [15] від однойменної швейцарської компанії пропонує масштабовані, точні та надійні рішення, що базуються на використанні штучного інтелекту для вимірювання миттєвих емоційних реакцій споживачів в онлайн та роздрібному торгових середовищах. nViso визначає емоційний стан користувача за методом Екмана і враховує нейтральне емоційне забарвлення.

Переваги ПЗ**:**

- точність визначення характеру емоції;
- швидкодія на належно високому рівні.

Недоліки ПЗ**:**

- відсутність пробної версії для тестування;
- висока вартість використання.

Вигляд вікна програми nViso наведено на рис. 2.

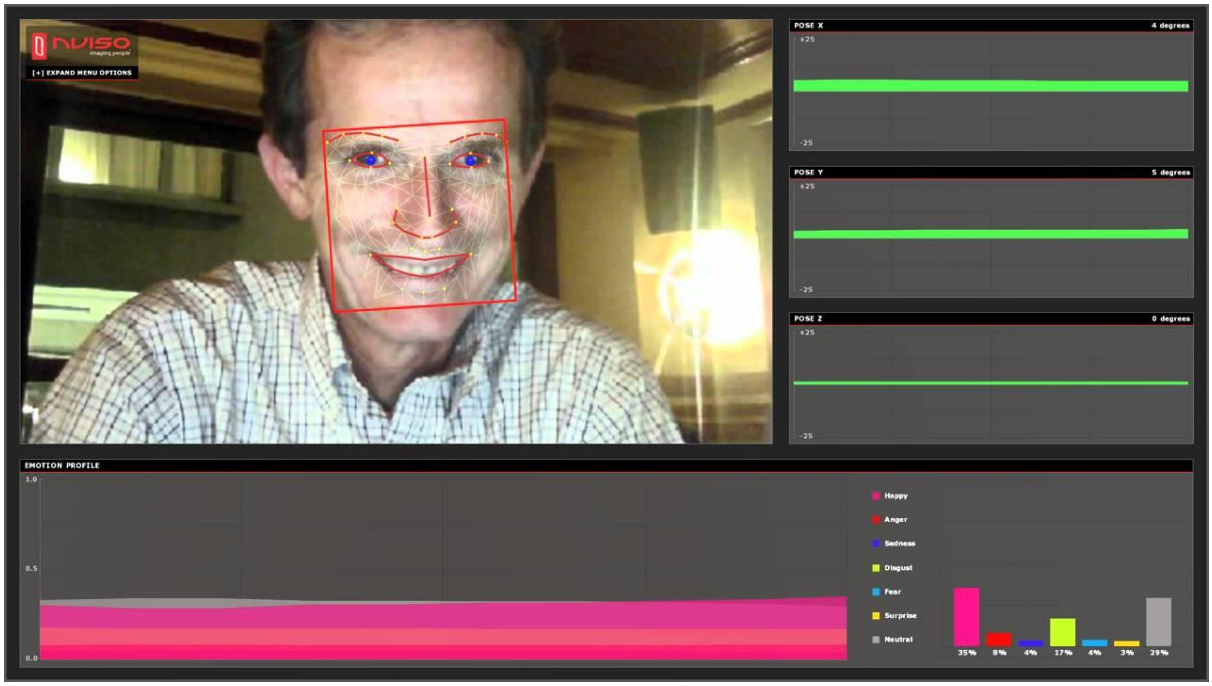


Рис. 2. Інтерфейс програми nViso [15]

Інші популярні застосунки для визначення емоцій

Існуючі API використовують розпізнавання обличчя, відслідковування положення очей та певні жестові сигнали для визначення настрою респондента. Зазвичай методи спираються на комбінацію з 7 базових емоцій, зважених за шкалою.

Emotient – застосунок для автоматизованого захоплення обличчя та виявлення емоційного забарвлення, що може бути інтегрований в веб-додатки або використовуватись для тестування. Окрім API, Emotient включає в себе аналітичний модуль з графіками та діаграмами. На сьогодні сервіс входить у склад компанії Apple та є недоступним [16].

Kairos Face Recognition API – приклад web-API, що дозволяє окремим розробникам чи бізнесам інтегрувати функціональні можливості розпізнавання емоцій у власні сервіси. Функціональність додатку включає такі опції: розпізнавання наявності обличчя, виявлення певного обличчя, пошук людини за даними її обличчя, визначення віку, статі та розпізнавання одразу кількох облич. Принцип роботи ґрунтується на побудові 3D-моделей за лицевими координатами і моніторингу їх положення у реальному часі [17].

Переваги ПЗ**:**

- якість вихідних результатів аналізу;
- наявність статистики, відображеної графічно;
- широкий спектр застосувань.

Недоліки ПЗ**:**

- відносно низька швидкодія через використання 3d-рушіїв та складних алгоритмів обробки;
- як наслідок не дуже підходить для динамічного аналізу емоційного забарвлення;
- відсутність простого способу інтегрування у веб-додаток – потрібні навички програмування;

- висока вартість використання додатку;
- недоступність через відсутність публічного API.

1.3. Актуальність розробки програмного застосунку

У сучасному медійному менеджменті як ніколи раніше власники інформаційних джерел неперервно конкурують за споживача. Згідно останніх даних аналітиків, близько 4 мільярдів людей користуються мережею інтернет. Останні прогнози показують, що інтернет є єдиною медіа платформою, в якої збільшується обсяг аудиторії. Особливо це актуально для молодих категорій населення. З часом ці зміни будуть стосуватись і більш дорослих користувачів. Охоплення традиційних медіа, як, наприклад, телебачення чи радіо, з року і рік не змінюється суттєво та часто зменшується. Тому акцент на розвитку інтернет-реклами є передовим та перспективним напрямком.

Головним ресурсом монетизації веб-сайтів наразі є реклама. Звичайний користувач може споглядати її у вигляді банерів, плакатів на сайтах, прихованої реклами у відео, в соцмережах тощо. Завдяки збільшенню різноманітності Інтернет-медіа і зміні поведінки користувачів перед постачальниками реклами постала проблема ефективного її впровадження [13]. Згідно з останніми спостереженнями, споживачі все більш скептично сприймають рекламні оголошення. Головна причина тому – засилля мережі неякісними і неактуальним для них рекламним контентом. Тому надання більш релевантної реклами стане для обох сторін фінансово вигідним чинником і так чи інакше позитивно впливатиме на репутацію медіа в очах користувачів.

Щоб якомога швидше отримувати на стороні постачальника актуальну реакцію на рекламний продукт, з-поміж багатьох відомих способів можна використати аналіз емоційного стану користувача. Емоційна реакція так чи інакше свідчить про характеристику та ступінь ставлення глядача до реклами в момент її перегляду, що значно прискорює

зворотній зв’язок між переглядачем веб-сайтів та рекламодавцем. Емоції – наріжний камінь реклами. Зазвичай зв'язок з певним брендом чи товаром споживач ініціює через емоційну реакцію. Донесення цієї інформації є важливим завданням.

1.4. Визначення вимог до програмного застосунку

В результаті проведеного дослідження було сформульовано такі функціональні вимоги до програмного застосунку для визначення емоційного забарвлення зображень:

- програмне забезпечення має реалізовувати клієнт-серверну архітектуру;
- має використовувати алгоритм розпізнавання облич;
- в реальному часі збирати дані з відеопотоку веб-камери та передавати їх на модуль обробки зображень;
- мати здатність роботи на сторонньому веб-сайті без потреби у залученні програміста;
- має відображати статистичні дані переглядів у окремому вікні в реальному часі;
- використовувати засоби комп’ютерного зору та машинного навчання для визначення кількісної і якісної характеристики емоційного забарвлення у такому діапазоні:
    - страх;
    - відраза;
    - нейтральний стан;
    - гнів;
    - щастя;
    - презирство.
- забезпечити збереження отриманих статистичних даних у БД.

Та такі додаткові вимоги:

- приведення масиву статистичних даних у зручний для опрацювання аналітиком формат;
- застосунок повинен мати негроміздкий UI для керування та відображення даних;

1.5. Висновки до розділу

Розробки в сфері штучного інтелекту розповсюджені у майже всіх сферах людської діяльності. Емоційний аналіз на сьогодні є важливим інструментом, що може використовуватись у багатьох галузях від медицини до банківської справи. Зокрема він може бути корисним для оцінки настрою і відношення споживача до наданої інформації про продукт. Ринок таких систем активно зростає – за результатами досліджень приріст сягає третини на рік. Такі показники виправдані, оскільки вже зараз програмне забезпечення для розпізнавання емоцій, поведінки, настрою за допомогою відео потоку з веб-камери чи іншого спеціалізованого устаткування, дістає широке застосування у медицині, сфері безпеки, при проведенні судових тяжб тощо.

Програмне забезпечення у вигляді веб-застосунку, що є метою даного дипломного проєкту буде використовуватись для відслідковування і оцінки емоційного стану глядача на будь-якому сайті. Важливим є надання можливості подання зібраних користувацьких даних у зручному для обробки аналітиком вигляді.

Огляд схожих за призначенням програмних додатків показав, що основною перешкодою на шляху їх використання є необхідність довгого налаштування та навичок програмування. В деяких випадках ПЗ недоступне за причин політики розробників.

Розроблюване програмне забезпечення має відповідати вимогам універсальності, зручності у користуванні та позбавляти користувача

необхідності витрачати час на додаткове налаштування, що є основною перевагою в порівнянні з альтернативними застосунками.

2. ОБҐРУНТУВАННЯ АРХІТЕКТУРИ ПРОГРАМНОГО ЗАСТОСУНКУ ТА ВИБОРУ ЗАСОБІВ РЕАЛІЗАЦІЇ

2.1. Обґрунтування вибору мови програмування

При виборі мови програмування для реалізації даного проєкту головним чином має враховуватись наявність і різноманіття інструментів для реалізації клієнт-серверної архітектури, клієнтської та серверної частини додатку. Такими інструментами слугуватимуть фреймворки та бібліотеки. Саме тому на етапі планування було розглянуто фреймворки і бібліотеки, створені для мов Java, Python та JavaScript – одних з найпопулярніших засобів для розробки веб-додатків.

2.1.1.Огляд фронтенд-технологій

React – відкрита JavaScript-бібліотека для побудови клієнтських інтерфейсів користувача, що дозволяє створювати великі за обсягом веб-застосунки, що оперують даними, які змінюються в часі без необхідності постійного перезавантаження сторінок. React добре поводить себе при масштабуванні додатків. В сукупності з бібліотекою Redux широко використовується для реалізації шаблону MVC. React розробляється Facebook та користується популярністю широкої спільноти розробників [19].

Переваги ПЗ:

- робота з віртуальною об’єктною моделлю документа(Virtual DOM);
- гнучкість та невибагливість до пам’яті;
- інтуїтивність побудови компонентів інтерфейсу;
- легкість при оптимізації пошукових систем.

Недоліки ПЗ:

- необхідний час на освоєння особливостей складових бібліотеки для їх правильної інтеграції;

- невпорядкованість і складність документації бібліотеки;

Angular – фронт-енд фреймворк з відкритим вихідним кодом, який розробляється під егідою компанії Google та, як і React, має велику спільноту користувачів і може застосовуватись для розробки веб- і мобільних додатків. Angular має зручні інструменти для тестування та відлагодження додатку, а інтуїтиивний API дозволяє будувати складні високопродуктивні системи [20].

Переваги ПЗ:

- швидка компіляція;
- мінімізовані ризики помилок;
- впровадження залежностей та модульності;
- зрозуміла детальна документація.

Недоліки ПЗ:

- відносно складний синтаксис;
- проблеми з міграцією між версіями.

VueJs – JavaScript фреймворк з відкритим вихідним кодом, призначений для проєктування користувацьких інтерфейсів та односторінкових додатків. Фреймворк оснований на архітектурним шаблоні MVVM(Model-View-ViewModel). Інтерфейс користувача базується на моделях даних через реактивне зв’язування даних. VueJS характеризуєтсья адаптивним підходом, який фокусуєтсья на декларативному рендерингу та створенні компонентів. Розширені можливості для складних додатків, такі як маршрутизація, управління станом та автоматизація можуть бути налаштовані через сторонні плагіни, наприклад, Nuxt.js [21].

Переваги ПЗ:

- ретельно прописана документація;
- підхід зв’язування даних за допомогою реактивної системи;
- достатня швидкодія для більшості веб-додатків;
- простота написання компонентів інтерфейсу;
- легкість в масштабуванні програмної системи.

Недоліки ПЗ:

- система рендерингу компонентів порівняно менш функціональна;
- невелика кількість сторонніх плагінів.

2.1.2. Огляд серверних технологій

NodeJS – серверна платформа для роботи з JavaScript, що базується на рушії V8 та асинхронному середовищі виконання, що базується на концепції циклу подій. NodeJS дозволяє створювати легко масштабовані веб-додатки. JavaScript в його складі використовується і для опрацювання даних на стороні сервера, і для обробки на стороні користувача [22].

Переваги:

- швидкість обробки запитів завдяки асинхронній моделі та циклу подій;
- неблокуючий асинхронний ввід та вивід даних;
- система модулів CommonJS;
- велика сукупність сторонніх пакетів NPM.

Недоліки:

- незручність при відлагодженні помилок;
- неефективність при обробці важких даних.

Spring – один з найбільш широко використовуваних фреймворків для розробки комерційних додатків, що забезпечує продуману модель програмування і конфігурації. В основі розробки лежить стек технологій Java EE і мова програмування Java. Spring може бути задіяний на всіх архітектурних шарах, використовуваних при розробці web-додатків [23].

Переваги:

- дозволяє з легкістю тестувати модулі вбудованими засобами;
- надає просту структуру при написанні класів;
- підтримує різні способи конфігурації.

Недоліки:

- має відносно довге налаштування;
- освоєння мови програмування Java накладає додаткові часові витрати.

Django – високорівневий веб-фреймворк, розроблений мовою Python. Підходить як для написання front-end частини, так і для back-end. Архітектура Django подібна до MVC, в Django це – MTV (Model – Template – View). Від самого початку фреймворк чудово підходить для створення інформаційних ресурсів, бо звільняє розробника від розробки системи керування вмістом. Вбудований адміністративний модуль дозволяє змінювати наповнення сайту, а також надає інтерфейс для розподілу прав між користувачами [24].

Переваги:

- швидкість розробки;
- легкість масштабування;
- захист від помилок, SQL-ін'єкцій та крос-сайтового скриптингу;
- наявність типових блоків- авторизації, підписки на розсилку тощо;
- вбудована функціональність з менеджменту контенту.

Недоліки:

- погано підходить для невеликих проєктів;
- немає підтримки веб-сокетів за замовчуванням;
- непередбачуваність поведінки деяких компонентів.

2.2. Обґрунтування вибору системи керування базами даних

PostgreSQL – обєктно-реляційна СКБД з відкритим вихідним кодом. Дані зберігаються в табличному вигляді, а таблиці звязуються за заданими правилами. Особливістю і перевагою над іншими реляційними SQL- СКБД є те, що PostgreSQL підтримує ряд користувацьких об'єктів та їх поведінки, включаючи функції, типи даних, домени та індекси [25].

Окрім цього СКБД може створювати, зберігати й діставати складні структури даних, такі як вкладені та складені конструкції. Також дана СКБД має обширний список типів даних, дозволяє будувати багатовимірні масиви, підтримує геометричні та специфіковані користувачем дані. PostgreSQL заслужено вважається надійною СКБД, адже в ній підтримуються властивості ACID.

MongoDB – документоорієнтована СКБД з відкритим кодом, представник типу NoSQL. баз даних Mongo не потребує опису схеми таблиць і зберігає записи в JSON-подібному форматі. MongoDB має достатньо зручну мову для написання запитів, дає змогу створювати індекси для збережених атрибутів, підтримує журналювання операцій зміни й додавання даних, а також може використовувати парадигму Map/Reduce, підтримує реплікацію і є стійкою до відмов [26].

2.3. Огляд засобів реалізації емоційного аналізу

Face-api – JavaScript-бібліотека, призначена для виконання широкого діапазону дій з обробки зображень як безпосередньо в браузері, та і на стороні сервера. Зокрема в ній представлені модулі для задач розпізнавання й аналізу емоційного забарвлення користувача безпосередньо у браузері. Дана бібліотека оформлена у вигляд і web-API. Функціональність базується на відкритій програмній бібліотеці для машинного навчання широкому спектру задач TensorFlow [26], що розроблена компанією Google для вирішення завдань тренування нейронних мереж, розробки широкофункціональних класифікаторів для виявлення і розпізнавання образів та кореляцій, досягаючи результатів, які можна порівняти з сприйняттям візуальних образів людським оком. [27].

Переваги:

- швидкодія обробки зображень;
- відсутність потреби в підготовці даних, оскільки для типової задачі розпізнавання TensorFlow має готові модулі;

- зручний спосіб взаємодії через API;
- відкритий вихідний код.

Недоліком даної бібліотеки можна вважати відносно погано деталізовану документацію, що виправдано тим, що це відкрита open-source розробка.

Keras – відкрита нейромережева бібліотека, написана мовою Python. Вона являє собою надбудову над фреймворками DeepLearning4j, TensorFlow та Theano і надає високорівневий інтерфейс над ними. Keras створено для уможливлення проведення швидких експериментів з мережами глибинного навчання, а її архітектура спроєктована у модульний і розширюваний спосіб. Keras містить значну кількість інструментів для спрощення роботи з зображеннями та текстом. Дана бібліотека має відкритий вихідний код та широко використовується як провідними IT-компаніями, так і ентузіастами [27].

Переваги:

- порівняно проста у використанні, бо значна частина інструментів для роботи з графікою вже наявна;
- дозволяє швидке створення мінімально працюючих прототипів;
- постійно вдосконалюється, додаючи нові модулі;
- має деталізовану документацію;
- містить багатий інструментарій для обробки зображень та відео.

Недоліками бібліотеки можна вважати дещо низьку швидкодію при використанні як основний компонент великих проєктів та особливості мови Python.

OpenCV – відкрита бібліотека функцій та алгоритмів комп’ютерного зору, обробки зображень та чисельних алгоритмів загального призначення.Вона надає засоби для обробки і аналізу вмісту зображень, у тому числі розпізнавання об’єктів на фотографіях(фігур людей чи облич) і для цього має понад 2500 алгоритмів машинного навчання і комп’ютерного

зору. Головний інтерфейс бібліотеки написаний на мові C++. Крім цього, реалізовано інтерфейси на мовах Python, Java, JavaScript та інших [28].

Переваги:

- велика база алгоритмів, що охоплює широкий спектр застосування;
- потужна підтримка і деталізована документація;
- наявність інтерфейсів для найпопулярніших мов програмування.

Недоліки:

- відносна важкість освоєння для виконання поставленої задачі;
- порівняно мала швидкодія для роботи в браузері.

2.4. Архітектура програмного застосунку

Архітектура програмної системи визначає зв’язки між окремими її компонентами та зовнішнім середовищем, спосіб і принципи їх реалізації. Найпоширенішими типами архітектур на сьогодні є монолітна та мікросервісна архітектури.

Мікросервісна архітектура

Мікросервісний стиль – це підхід, коли єдиний додаток створюється як сукупність невеликих та незалежних, не тісно зв’язаних сервісів, що обмінюються даними за допомогою механізмів як-то HTTP, gRPC, AMQP. Кожен сервіс відповідальний за певну бізнес-логіку та розгортається незалежно з використанням повністю автоматизованого середовища. Існує абсолютний мінімум централізованого управління цими сервісами. Самі по собі сервіси можуть бути реалізовані на різних мовах програмування і використовувати різні технології зберігання даних[29].

Цей стиль розробки отримав розповсюдження завдяки поширенню практик гнучкої розробки. Коли важливо забезпечити швидке розростання системи, вдосконалення модулів для підтримання конкурентоспроможності продукту і своєчасну доставку змін, особливо у великих корпоративних програмних системах, – він має значні переваги.

Переваги мікросервісної архітектури:

- краща архітектурна організація, оскільки кожен модуль призначений для виконання окремої задачі;
- незвязані сервіси легше переналаштувати для обслуговування різних потреб;
- можливість ізоляції проблем на одному сервісі не впливаючи на роботу всієї системи;
- незалежне масштабування окремих компонентів.

Недоліки мікросервісної архітектури випливають з її означення.

- мережеві затримки, що можуть виникати між окремими модулями при відділеній взаємодії;
- складність операційної підтримки, необхідність залучення DevOps-інженерів та розгортання автоматичного моніторингу;
- відносна складність тестування;
- необхідність узгоджувати формати повідомлень між мікросервісами.

Монолітна архітектура

Монолітна архітектура визначає, що застосунок являє собою один загальний модуль, ресурси і пам’ять якого будуть використовуватись спільно усіма процесами всередині. Вона дозволяє отримати більшу швидкодію, оскільки прямий доступ до даних у пам’яті працює швидше ніж комунікація через посередній модуль між окремими мікросервісними процесами.

При монолітності архітектури існують такі переваги для розробника:

- модулі простіше реалізовувати;
- краще підходить для невеликих додатків;
- простіше приєнання типової функціональності;
- монолітна система в майбутньому простіша в масштабуванні.

Недоліки монолітної архітектури:

- при різкому збільшенні обсягу проєкту стає складніше ізолювати окремі компоненти;
- розуміння коду монолітної структури вимагає зазвичай більших зусиль у розробника;
- складніше масштабування, бо при необхідності зміни одної частини постає потреба змінювати всі з нею пов’язані.

Оскільки розроблюваний у даному дипломному проєкті веб-застосунок є порівняно невеликим за обсягом і для впровадження функціональності мережеві затримки є критичними, то для його реалізації було обрано клієнт-серверну монолітну архітектуру.

Веб-сервер являє собою один з основних компонентів системи, що призначений обробляти вхідні дані користувача, такі як дані при авторизації, статистичні дані, отримані від клієнтської частини додатку, запити користувача при взаємодії з елементами інтерфейсу програми та об’єднувати клієнтську частину додатку з даними, що зберігаються в віддаленій базі даних і обслуговуються системою керування базами даних. На рис. 3 зображена загальна структура програмного застосунку.

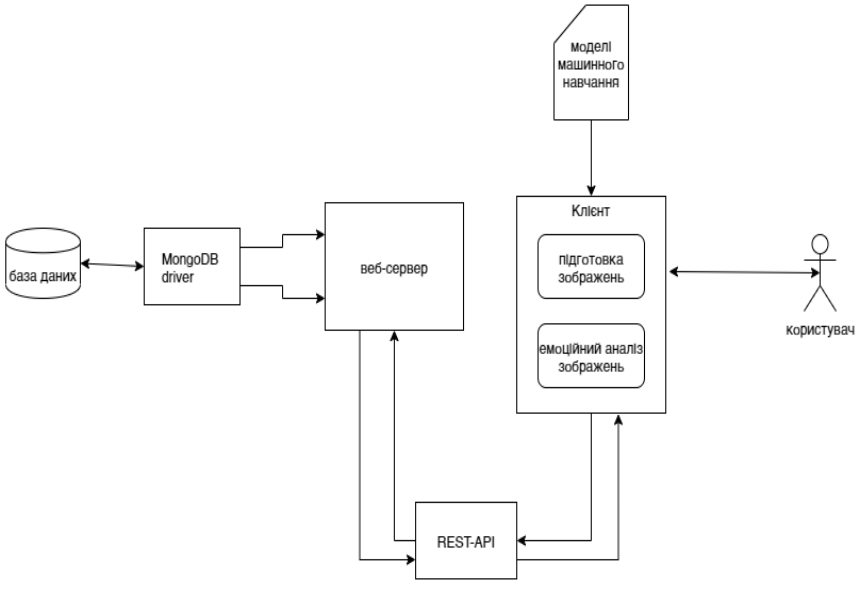


Рис. 3. Структурна схема сервісу

Функціональність, що відповідає за розпізнавання та аналіз емоційного забарвлення обличчя користувача працює на стороні клієнта, що дозволяє отримувати зворотній зв’язок набагато швидше, аніж це було б при обміні даними з віддаленим сервером.

Web-Server – серверна частина застосунку, розроблена з використанням технологій мови JavaScript, таких як NodeJS та ExpressJS, а також множини сторонніх модулів. Дана зв’язка дозволяє у короткий термін розробляти продуктивні і високошвідкісні веб-додатки. Крім цього, вона особливо добре підходить для невеликих проєктів. Веб-сервер складається с модулів, що відповідають за представлення моделей даних з БД, їх серіалізацію у формат JSON строки, обробку запитів за допомогою API та надсилання веб-сторінок та статичних файлів на клієнтську частину.

2.5. Висновки до розділу

В даному розділі було розглянуто архітектуру програмного застосунку, ряд інструментів для його розробки та окреслено їх переваги й недоліки. Зокрема було оглянуто бібліотеки для реалізації розпізнавання і оцінки емоційного забарвлення зображень. Отримана інформація необхідна для визначення необхідного набору засобів, які буде застосовано при розробці програмного застосунку для аналізу емоційного забарвлення зображень.

Мовою програмування для даного проєкту було обрано JavaScript.

Для розробки веб-застосунку були обрані наступні інструменти:

- для серверної частини – NodeJS + Express;
- для клієнтської частини – React.

Зв’язка NodeJs з ExpressJs надає змогу писати високонавантажувані серверні частини та API і особливо добре підходить для невеликих проєктів. React Js – бібліотека, яка має багаті функціональні можливості з рендерингу, зручний підхід до побудови компонентів та широку базу шаблонів UI.

В якості системи керування базами даних було обрано MongoDB через зручність використання JSON-нотації, інтеграції з мовою програмування JavaScript, документну модель даних, легкість масштабування і швидкодію.

Для реалізації функцій аналізу емоційного забарвлення зображень було вибрано бібліотеку Face-API через швидкість роботи в браузері та зручний програмний інтерфейс.

3. РОЗРОБЛЕННЯ ПРОГРАМНОГО ЗАСТОСУНКУ

3.1. Архітектура бази даних

Для зберігання даних, зібраних під час сесій користувачів було обрано СКБД MongoDB. Для подання моделей даних у зручному для опрацювання в коді вигляді вибір зроблено на користь неї, адже MongoDB не потребує чіткого опису схеми таблиць і має певні переваги як швидких і потужних реляційних БД, так і баз даних, що зберігають дані в форматі ключ-значення. Вона добре підходить для роботи в асинхронному середовищі NodeJS та дозволяє визначати об’єкти зі строго-типізованою схемою, що відповідає документу MongoDB.

Окрім цього, MongoDB вирізняється з-поміж інших СКБД повною підтримкою JSON-подібного формату зберігання документів, наявністю функціональних можливостей з журналювання подій додавання, зміни і видалення даних з таблиць, широким асортиментом інструментів для агрегації даних та побудови стійких до збоїв конфігурацій БД.
Додатково MongoDB має широкі функціональні можливості з маніпуляції даними, валідації даних, конвертації типів даних тощо.

Структура бази даних

Дані застосунку являють собою особисті дані користувачів та дані, що стосуються перегляду веб-сайтів.

Отже, створено 3 таблиці:

- users;
- userSessions;
- sessionVisits.

Users

Це таблиця, що містить список особистих даних усіх користувачів додатку, що дали дозвіл на використання їх зображень у цілях аналізу емоційного стану. Зміст та перелік полів таблиці представлено у таблиці 1.

Таблиця 1

Users

Назва поля	Тип даних
id	autoincrement
username	string
email	string
passhash	string

Дана таблиця містить наступні поля:

- id – унікальний ідентифікатор користувача, що автоматично генерується системою при додаванні;
- username- імя користувача;
- email – пошта;
- passhash – пароль в хешованому вигляді.

UserSessions

Таблиця, що містить інформацію про сесію користувача, а саме поля ідентифікатора сесії, момент початку сесії та кінця сесії та відповідні їм типи даних.

Таблиця 2

Users

Назва поля	Тип даних
Session_id	Autoincrement
Time_start	Datetime
Time_end	Datetime

Таблиця UserSessions містить такі поля:

- session_id – унікальний ідентифікатор сесії;
- time_start – початок сесії;
- time_end – закінчення сесії.

SessionVisits

Таблиця, що містить дані спостережень за емоціями користувача від перегляду перного веб-ресурсу.

Таблиця 3

SessionVisits

Назва поля	Тип даних
Session_id	String
Site_url	String
time_eval	Datetime
emotion	Object

У таблиці містяться наступні поля:

- session_id – ідентифікатор сесії;
- site_url – URL-адреса веб-сайту;
- time_eval – час, коли проведено оцінку емоційного забарвлення;
- emotion – набір даних, що характеризують емоцію в процентному співвідношенні.

3.2. Реалізація аналізу емоційного забарвлення зображень

Функціональність з обробки зображень, розпізнавання на них облич та визначення емоцій основана на бібліотеці face-api-js. Бібліотека побудована поверх відомої відкритої платформи для машинного навчання TensorFlow, що надає бібліотеку готових алгоритмів чисельних обчислень, реалізованих через графи потоків даних (data flow graphs). Вузли в таких графах реалізують математичні операції або точки входу/виходу, тоді як ребра графа представляють багатовимірні масиви даних (тензори), які перетікають між вузлами. Вузли можуть бути закріплені за обчислювальними пристроями і виконуватися асинхронно, паралельно обробляючи разом все підходящі до них тензори, що дозволяє організувати

одночасну роботу вузлів нейронної мережі за аналогією з одночасною активацією нейронів в мозку.

Виявлення обличчя на зображенні

Перший етап на шляху до визначення емоційного забарвлення полягає в знаходженні на зображенні обличчя ключових точок. Для цього був використаний модуль бібліотеки tinyFaceDetector – ефективний детектор обличчя в режимі реального часу, базований на згортковій нейромережі [30] Tiny Yolo V2. Нейромережева модель детектора опрацювала 14 тисяч зразків зображень людського обличчя, позначених мітками. Модель була навчена передбачати обмежувальні рамки, які покривають ключовими точками всю поверхню обличчя. Вона є надзвичайно мобільною та зручною для браузерних додатків, оптимізованою для роботи на пристроях з обмеженими ресурсами, такими як смартфони.

Розпізнавання виразу обличчя

Після того, як обличчя локалізовано за допомогою масиву контрольних точок, в дію вступає модель (Face Expression Recognition Model). Модель має розмір 310 Кб, що робить її зручною для використання у легких додатках. Вона використовує глибоко відокремлювані згортки і щільно з’єднані блоки. Модель проходила підготовку на різноманітних зображеннях облич з загальнодоступних наборів даних, а також зображеннях, зібраних з мережі.

Оскільки моделі оптимізовані для роботи в непотужному середовищі, точність дещо зменшена порівняно з повноцінними моделями. Також варто зазначити, що на точність можуть впливати перешкоди між обличчям людини та камерою, наприклад, бліки від лінз окулярів тощо.

Розроблений програмний застосунок здатний розпізнавати сім типів емоційного забарвлення. Опис наявних типів наведено у таблиці 4.

Таблиця 4

Типи емоційного забарвлення

Характер забарвлення	типи
позитивний	happy – вияв щастя surprised - здивування
негативний	sad - сум angry - злість fearful - переляк disgusted – вияв відрази
нейтральний	neutral – нейтральний стан

3.3. Особливості реалізації

Програмний застосунок виконано у вигляді веб-додатку та реалізовано в монолітному стилі на основі клієнт-серверної архітектури. Він містить модулі, серед яких такі:

- серверний модуль;
- модуль графічного інтерфейсу;
- модуль обробки зображень.

Серверний модуль

Серверний модуль відповідає за створення зв’язку між клієнтською частиною застосунку та базою даних, а також у генерації сторінок авторизації та аналітики.

Для реалізації серверної частини було використано асинхронне середовище виконання Javascript – NodeJS, фреймворк для розробки веб-додатків Express JS, що був використаний для організації REST-API точок прийому запитів на стороні сервера, та інші сторонні модулі npm.

Вікно авторизації

Дане вікно призначене для авторизації адміністратора системи для подальшого перегляду аналітичних даних, що були отримані при взаємодії користувача з системою. Авторизація відбувається шляхом введення логіна

та пароля і відповідні поля. Передбачено клієнтську валідацію для запобігання введення некоректних даних. Вікно витримано у загальному стилі програми.

Вигляд вікна авторизації проілюстровано на рис. 4.

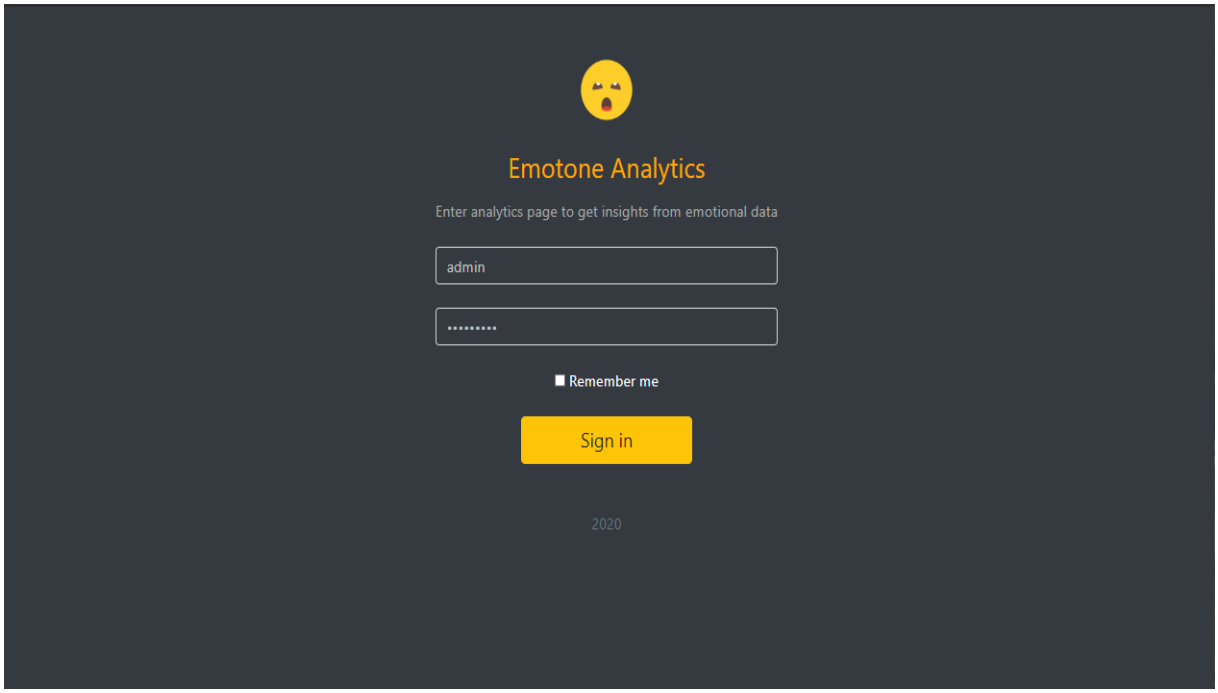


Рис. 4. Вікно авторизації

Вікно аналітики

У вікні аналітики система відображає такі елементи як графіки, діаграми, таблиці та кнопки, що відповідають за функції оновлення і збереження даних. Для прикладу, на показаному нижче вікні аналітики виводиться такі дані як кругова діаграма співвідношення між різними емоціями та лінійний графік розподілу емоцій протягом періоду часу вимірювань, який задано в програмі. На таблиці у нижній частині вікна показано кількість задоволених, незадоволених та загальне число користувачів.

При активації кнопки “update graphs” на серверний модуль відправляється запит для оновлення аналітики. Кнопка “Export data”

дозволяє зберегти отримані дані. Графіки та діаграми побудовано за допомогою бібліотеки ChartJS.

Вигляд вікна аналітики наведено на рис. 5.

Рис. 5. Вікно аналітики

Модуль графічного інтерфейсу

У даному програмному застосунку для побудови адаптивного користувацького інтерфейсу використано класичні технології для розмітки та стилізації веб-сторінок, такі як HTML та CSS, а також бібліотека компонентів бібліотеки Bootstrap. Для додатку розроблено три вікна – головне вікно, вікно авторизації та вікно аналітики.

Після запуску програми користувача зустрічає головне вікно, що містить елементи керування відео потоком з веб-камери, вікно для виведення відео, органи навігації та посилання на інформацію про проєкт. Користувачу надано доступ до перегляду даних про емоційний стан та параметри відео потоку, такі як частота оновлення та частота кадрів за секунду відтворення відео. За потреби користувач може змінити роздільну

здатність потокового відео та мінімальну точність, з якою програма буде аналізувати зображення. Від цього залежить навантаження на ресурси пристрою.

Одразу після запуску користувач отримує запит на отримання програмою дозволу вести запис відео. Після надання доступу у головне вікно над органами керування транслюється відео потік.

Через мінімальну кількість часу система завантажує моделі машинного навчання і проводить обробку кадрів з відео. Окремо завантажуються моделі, попередньо натреновані на визначення рис обличчя, меж обличчя та виразу обличчя. Порядок їх завантаження є чітким – спочатку завантажується модель, що призначена для визначення на растровому зображенні об’єкта, схожого на обличчя. Ця модель використовувала для навчання тисячі зображень людей з мережі. Наступною моделлю в черзі є модель для побудови сітки контрольних точок. За контрольними точками наступним є співставлення співвідношень між ними з дескрипторами моделі, що призначена розпізнавати вирази обличчя. Для даного дипломного проєкту їх діапазон було обмежено до семи у варіаціях позитивних, негативних та нейтрального стану. Поверх вікна відео виводиться область захоплення обличчя та контрольні точки, визначені алгоритмами. За потреби користувач може приховати їх, натиснувши на відповідні елементи інтерфейсу під відео.

Приклад вигляду головного вікна програмного застосунку наведено нижче на рис. 6.

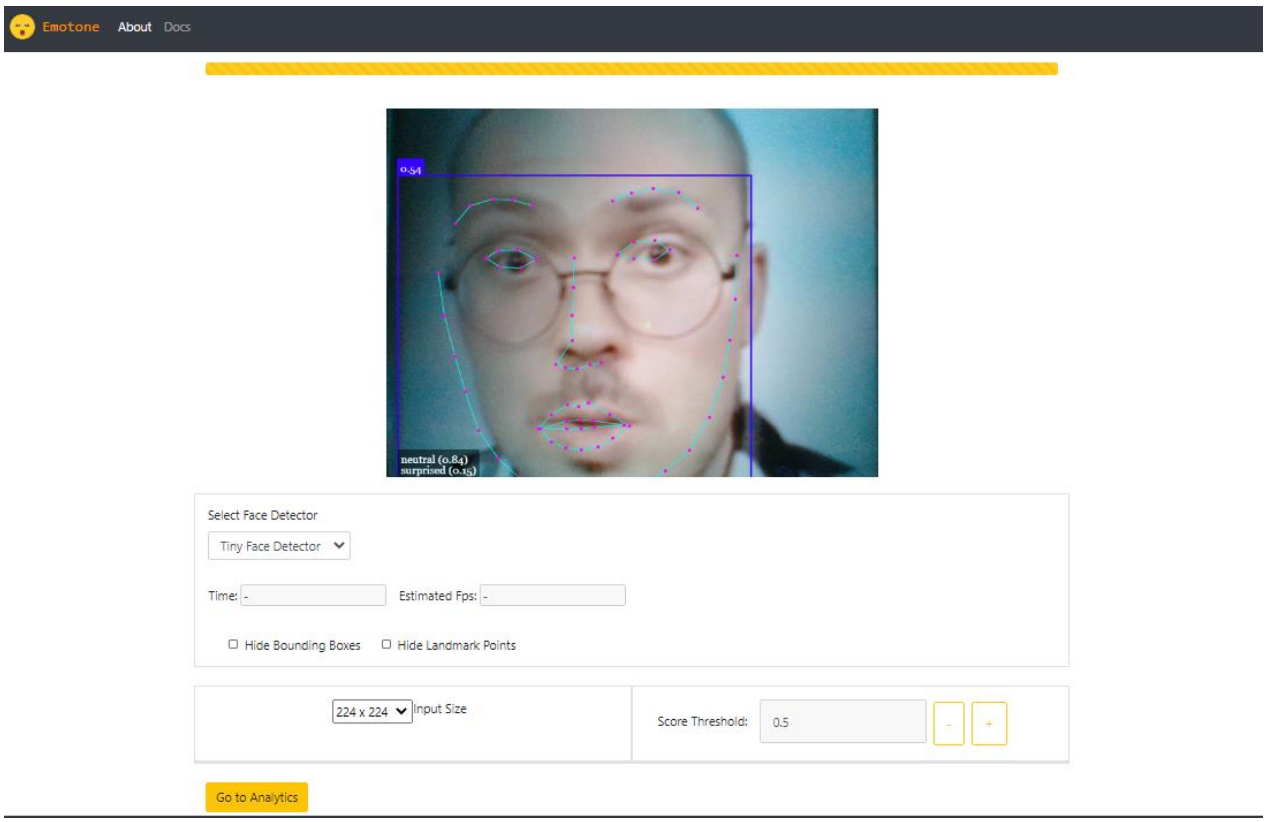


Рис. 6. Головне вікно програми

Модуль обробки зображень

Модуль обробки зображень являє собою набір функцій, що відповідають за прийом графічної інформації з відеопотоку камери, передачу отриманих даних на обробку засобами face-api, ретрансляцію виданої системою оцінки вхідних зображень(кадрів) у вигляді набору емоційних характеристик у відсотковому співвідношенні. Як вказано в описі до алгоритмів визначення емоційного забарвлення зображень, основою для вирішення цієї задачі є побудова контрольних точок та використання моделей машинного навчання, що підвантажуються на етапі запуску програми.

Приклади результатів обробки представлені на рисунках нижче (рис. 7, 8, 9).

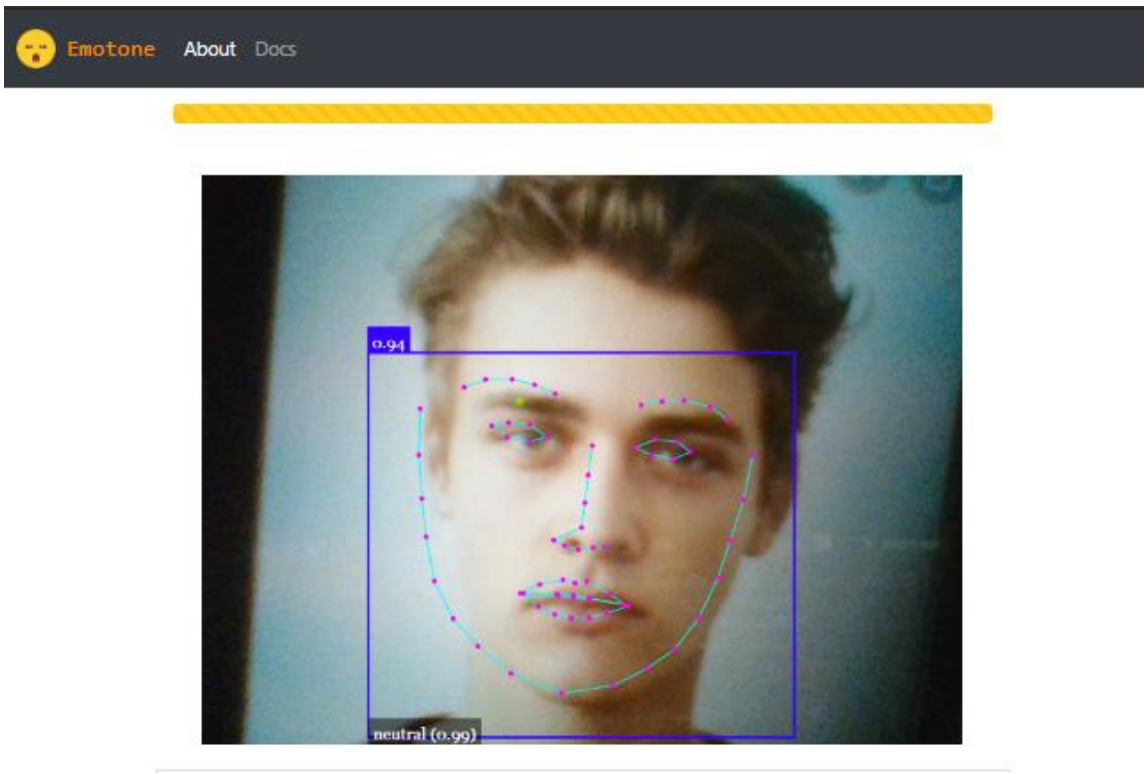


Рис. 7. Результат визначення емоційного забарвлення зображень

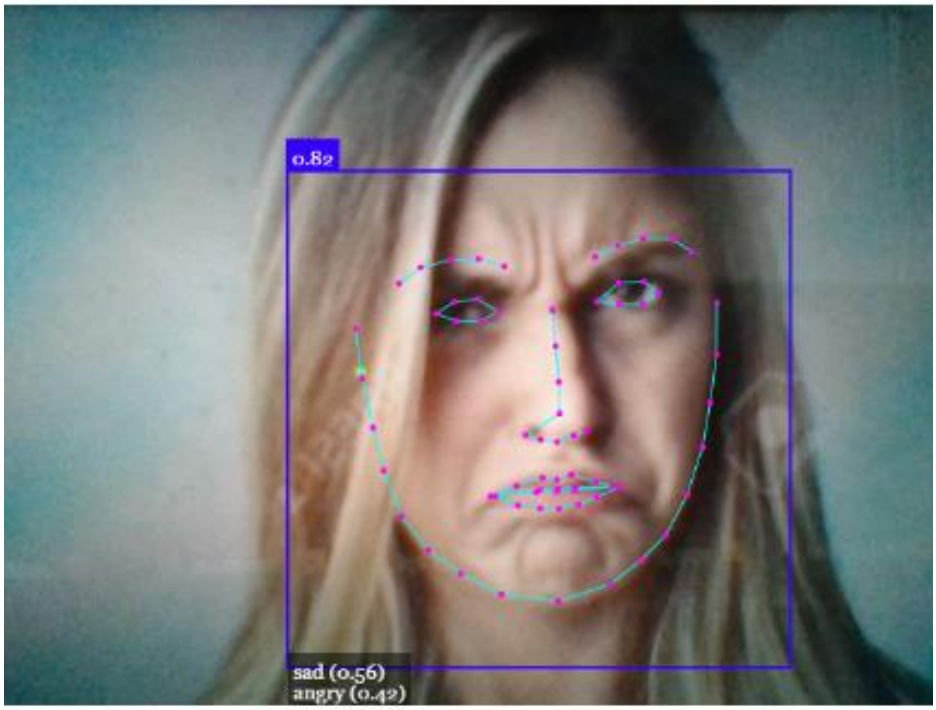


Рис. 8. Результат визначення емоційного забарвлення зображень

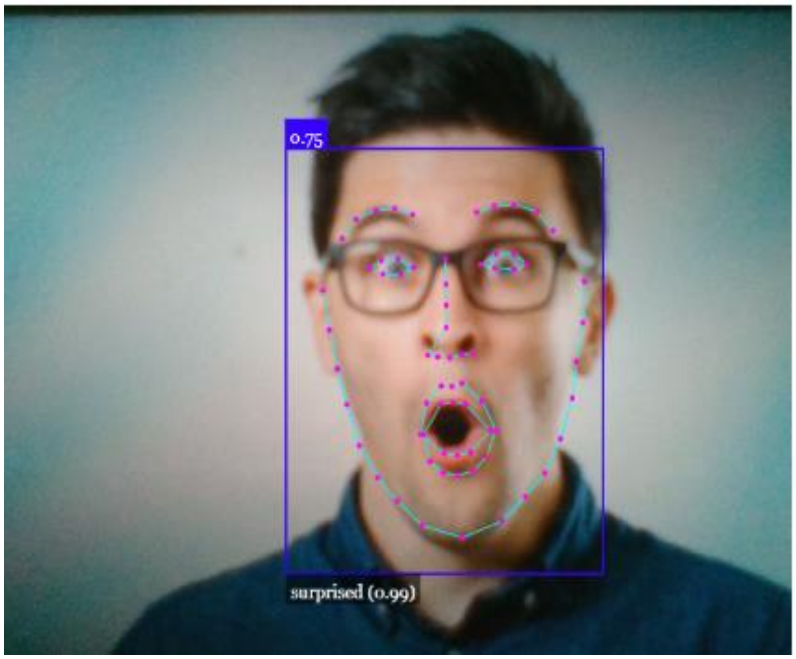


Рис. 9. Результат визначення емоційного забарвлення зображень

3.4. Висновки до розділу

В даному розділі відображено структуру бази даних програмного додатку, описано типи та значення полів таблиць СКБД MongoDB, вибір якої був обґрунтований для забезпечення надійного і зручного зберігання користувацьких даних. Крім цього було вказано види емоційного забарвлення для аналізу та наочно проілюстровано приклади виведення результатів роботи програмного застосунку за різних вхідних умов.

Окрім цього, було описано підхід до реалізації основної частини додатку – функціональності з розпізнавання та визначення кількісної і якісної характеристики емоційного забарвлення зображення обличчя користувача. Для цього у даному розділі було наведено загальний алгоритм роботи з визначення емоційного стану, засоби бібліотеки face-api, що використовуються для досягнення цієї мети. У даному розділі описано основні модулі програмного застосунку, такі як модуль обробки зображень, відповідальний за визначення емоційного забарвлення, модуль графічного інтерфейсу, призначений для відображення елементів керування додатком та серверний модуль, що реалізує точки доступу, взаємодіє з базою даних і

є основним рушієм даного застосунку. Також у розділі розглянуто вікна додатку, а саме головне вікно, вікно авторизації та вікно аналітики.

4. АНАЛІЗ РОЗРОБЛЕНОЇ СИСТЕМИ

4.1. Аналіз розробленого програмного застосунку

У першому розділі дипломного проєкту було розглянуто властивості аналогів розробленого програмного застосунку. Розгляд аналогів відбувався навколо їх позитивних та негативних сторін для вирішення конкретної задачі – використання емоційного забарвлення зображень для спостереження за реакцією користувачів. В ході дослідження було виявлено чимало спільних рис з нині наявними альтернативами, проте основним недоліком було виявлено відносні складність використання, потреба у налаштуванні спеціалістом чи недоступність за інших причин.

Отже, розробка проводилась з урахуванням усіх висунутих вимог, що випливали з необхідності отримати інструмент, що буде конкурентоспроможним на ринку таких застосунків і не матиме недоліків вищерозглянутих аналогів.

Аналіз розробленого застосунку дає підстави стверджувати, що він є актуальним і таким, що відповідає усім вищезгаданим вимогам, рішенням, а саме:

- реалізація розподілу прав доступу по ролях користувачів;
- реалізація функцій розпізнавання і визначення емоційного забарвлення зображень людини в режимі реального часу;
- подання даних з БД у зручному для подальшого аналізу форматі;
- оформлення елементів графічного інтерфейсу у зручному стилі;
- забезпечена швидкодія на малопотужних пристроях;
- відсутня необхідність додаткових навичок для впровадження додатку на сайт.

4.2. Тестування програмного застосунку

Тестування програмного застосунку є одним з важливих етапів в процесі розробки. В широкому розумінні тестування – одна з технік

контролю якості, яка включає планування, складення тестів, безпосереднє виконання і аналіз отриманих результатів. Метою тестування програмного додатка є отримання інформації щодо якості програмного продукту, відповідності поставленим вимогам і реально реалізованої функціональності. Процес перевірки відповідності виконується шляхом спостереження за роботою додатку на обмеженому наборі тестів. Вчасне тестування дозволяє запобігти непередбачуваним проблемам, з якими може зіштовхнутись потенційний користувач.

Після реалізації кожної окремої функції та модуля програмного застосунку його фукнціональність піддавалась проміжному тестуванню на наборі критичних тестових випадків. Після цього функціональність тестувалася безпосередньо у режимі роботи з додатком. Для проведення процесу тестування було обрано ручний метод. Тестування виконувалось розробником даного програмного забезпечення. Загалом, було виконане тестування серверної частини на коректність обміну даними, клієнтської частини на правильність відображення сторінок та статики, модуля емоційного аналізу як частини клієнтської сторони на швидкодію обробки зображень та бази даних на захист від ін’єкцій та перевірку коректності подання даних у таблицях СКБД. Кінцева версія додатку була надана на тестування реальним користувачам для отримання подальших рекомендацій щодо вдосконалення програмного додатку.

Під час тестування було виявлено й усунуто чимало недоліків різного ступеня критичності, наприклад:

- неоптимізований порядок виконання операцій, що сповільнювало запуск програми;
- конфлікт між сторонніми модулями системи;
- відправлення даних в БД в некоректному форматі.

Відповідно до цього було здійснено реорганізацію частин коду, що містили помилки й недоліки. Деякі з них були виявлені на етапі ручного

тестування під час розробки, інші – під час користування реальними людьми.

4.3. Пропозиції щодо поліпшення застосунку

Отриманий в процесі розробки програмний застосунок відповідає усім необхідним критеріям дипломного проєкту, а саме меті, темі та завданню проєкту. Функціонально, щоправда, він є мінімально життєздатним продуктом, який призначений для демонстрації можливостей розпізнавання емоційного забарвлення зображень людини та для втілення цієї мети містить лише необхідну кількість модулів, які може бути використано у подальшому розвитку для побудови більш складного застосунку для використання у реальних ситуаціях. Можливостей додатку на даному етапі достатньо щоб оцінити конкурентоспроможність та отримати достатню кількість відгуків від реальних користувачів щодо наявних мінусів реалізації функціональних та нефункціональних аспектів застосунку.

В ході опитування реальних користувачів було здійснене анкетування і як результат отримано перелік пропозицій щодо вдосконалення і розвитку програмного затосунку у наступних версіях:

- додавання реєстрації та авторизації за допомогою Google та Facebook;
- створення мобільної версії додатку;
- інтеграція додатку зі сторонніми рекламними API;
- поліпшення швидкодії модуля обробки зображень при повільному Інтернет-з’єднанні;
- виконання додатку у вигляді розширення для брузерів;
- додавання можливості отримання даних про користування сайтами у вигляді інфографіки на вказану пошту;

- додавання елементів швидкого зворотнього зв’язку з розробниками;
- вдосконалення зовнішнього вигляду і зміна навігації по вікну адмністратора.

Отже, прийняті пропозиції було взято до уваги і буде враховано при вдосконаленні системи у наступних випусках програмного застосунку.

4.4. Висновки до розділу

У даному розділі було проведено аналіз процесу розробки та методики тестування розробленого програмного веб-застосунку.

Впродовж всього часу розробки програмного забезпечення тестування роботи розроблених функціональних можливостей застосунку відбувалось ручним способом, при якому виявлялись проблемні сторони та вносилися необхідні зміни. Остаточну версію застосунку було надано для тестування реальним користувачам. Тестування показало, що програмний застосунок не позбавлений недоліків, які проявляються в неочікуваному сповільненні роботи застосунку, некоректності відображення даних у БД та конфліктах між сторонніми модулями системи. Отримані звіти користувачів було проаналізовано, а помилки у короткий термін виправлено внесенням змін у програму застосунку. Крім цього, від користувачів було отримано пропозиції щодо вдосконалення програмного застосунку у наступних версіях, переважна частина з яких стосувалась поліпшення швидкодії та стабільності роботи програми. Всі пропозиції було враховано і прийнято до уваги.

При прямому порівнянні властивостей даного програмного застосунку з існуючими рішеннями, що спеціалізуються на аналізі людських емоцій за зображеннями облич людей, було зроблено висновок, що розроблене програмне забезпечення має потенційно ширшу область застосування завдяки своїй універсальності, що робить його унікальним і більш конкурентним рішенням серед подібних додатків.

Підсумовуючи вищесказане можна вважати, що завдання і мета розробки даного програмного застосунку досягнуті відповідно до сформованих цілей та вимог.

ВИСНОВКИ

Метою даного дипломного проєкту було розроблення програмного застосунку, який призначений для відслідковування емоційного стану користувача за допомогою засобів аналізу емоційного забарвлення обличчя. Це, для прикладу, може допомогти постачальникам продукту, що рекламується у коротший термін та ефективніший спосіб розміщувати рекламні оголошення, адже дасть змогу їм отримувати зворотній зв’язок набагато швидше. Це, в свою чергу, зменшить фінансові витрати на маркетингові дослідження.

В попередніх розділах було окреслено існуючі проблеми та розглянуто представлені існуючі системи й застосунки, розглянуто їх переваги й недоліки. На основі цього було визначено ряд функціональних та нефункціональних вимог до розроблюваного програмного забезпечення.

В результаті проведеного аналізу проблем і вимог цільового програмного застосунку було вирішено реалізувати його у вигляді веб-додатку на основі клієнт-серверної монолітної архітектури мовою JavaScript.

Після порівняння засобів розробки було обрано для створення серверної частини обрати фреймворк NodeJS та ExpressJS, для клієнтської частини застосунку було обрано бібліотеку React.

В ролі системи керування базами даних виступає документна СКБД MongoDB, завдяки її способу взаємодії та подання даних, ACID-сумісності та надійності при високих навантаженнях.

В результаті створено програмний застосунок для аналізу емоційного забарвлення зображень обличчя користувача, який має в собі наступну функціональність, що відповідає висунутим вимогам:

- передбачення у системі 2 типів ролей користувачів: адміністратор та глядач;

- реалізація розпізнавання обличчя та оцінка емоційного забарвлення переглядача веб-сайту;
- збір масиву емоційних даних користувачів сайтів;
- подання масиву отриманих даних у зручному для аналізу вигляді;
- підтримання рівня швидкодії на належному рівні;
- негроміздкий мінімалістичний UI.

Розробку програмного забезпечення було виконано в повному обсязі. Програмний застосунок відповідає поставленим вимогам. Тестування проведено згідно з затвердженою методикою тестування.

СПИСОК ВИКОРИСТАНИХ ЛІТЕРАТУРНИХ ДЖЕРЕЛ

1. Database management system: [Електронний ресурс]. Режим доступу до ресурсу: https://searchsqlserver.techtarget.com/definition/database-management-system
2. User Interface: [Електронний ресурс]. Режим доступу до ресурсу: https://en.wikipedia.org/wiki/User_interface
3. MVC Design Pattern: [Електронний ресурс]. Режим доступу до ресурсу: https://www.geeksforgeeks.org/mvc-design-pattern/
4. HTTP overview: [Електронний ресурс]. Режим доступу до ресурсу: https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview
5. JSON – JavaScript Object Notation: [Електронний ресурс]. Режим доступу до ресурсу: https://www.json.org/json-en.html
6. Application Programming Interface: [Електронний ресурс]. Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Application_programming_interface
7. Jan Erik Solem. Programming Computer Vision with Python: Tools And Algorithms For Analyzing Images [Текст]. / Jan Erik Solem– 1st edition – Packt Publishing, 2019 – 296 p.
8. Kai Sasaki. Hands-On Machine Learning with TensorFlow.js: A guide to building ML applications integrated with web technology using the TensorFlow.js library [Текст]. / Kai Sasaki – 1st edition – O'Reilly Media, 2012 – 256 p.
9. Stuart Russell, Peter Norwig. Artificial Intelligence: A Modern Approach [Текст]. / Stuart Russell, Peter Norwig – 4th edition – Pearson, 2010 – 1152 p.
10. Emotion Detection and Recognition Market: [Електронний ресурс]. Режим доступу до ресурсу: https://www.marketsandmarkets.com/Market-Reports/emotion-detection-recognition-market-23376176.html

11. MindWave Brain Sensing Headset: [Електронний ресурс]. Режим доступу до ресурсу: https://store.neurosky.com/pages/mindwave
12. FaceReader : [Електронний ресурс]. Режим доступу до ресурсу: https://www.noldus.com/facereader
13. Новаківський, І.І., Любомудрова, Л.С. Оцінювання ефективності Internet-реклами [Електронний ресурс] – 2011 – Режим доступу до ресурсу: http://vlp.com.ua/files/24_26.pdf – (17.05.2020).
14. EmoDetect: [Електронний ресурс]. Режим доступу до ресурсу: https://emodetect.ru/
15. nViso application: [Електронний ресурс]. Режим доступу до ресурсу: https://www.nviso.ai/en
16. Emotient: [Електронний ресурс]. Режим доступу до ресурсу: https://www.crunchbase.com/organization/emotient
17. React vs Angular vs Vue.js — What to choose in 2020: [Електронний ресурс]. Режим доступу до ресурсу: https://medium.com/@TechMagic/reactjs-vs-angular5-vs-vue-js-what-to-choose-in-2018-b91e028fa91d
18. Angular: [Електронний ресурс]. Режим доступу до ресурсу: https://angular.io/
19. Getting Started with Vue – An Overview and Walkthrough: [Електронний ресурс]. Режим доступу до ресурсу: https://www.taniarascia.com/getting-started-with-vue/
20. NodeJS: [Електронний ресурс]. Режим доступу до ресурсу: https://nodejs.org/en/
21. Understanding the Basics of Spring vs. Spring Boot: [Електронний ресурс] Режим доступу до ресурсу: https://dzone.com/articles/understanding-the-basics-of-spring-vs-spring-boot
22. PostgreSQL Overview: [Електронний ресурс]. Режим доступу до ресурсу: https://www.postgresql.org/docs/12/plpgsql-overview.html

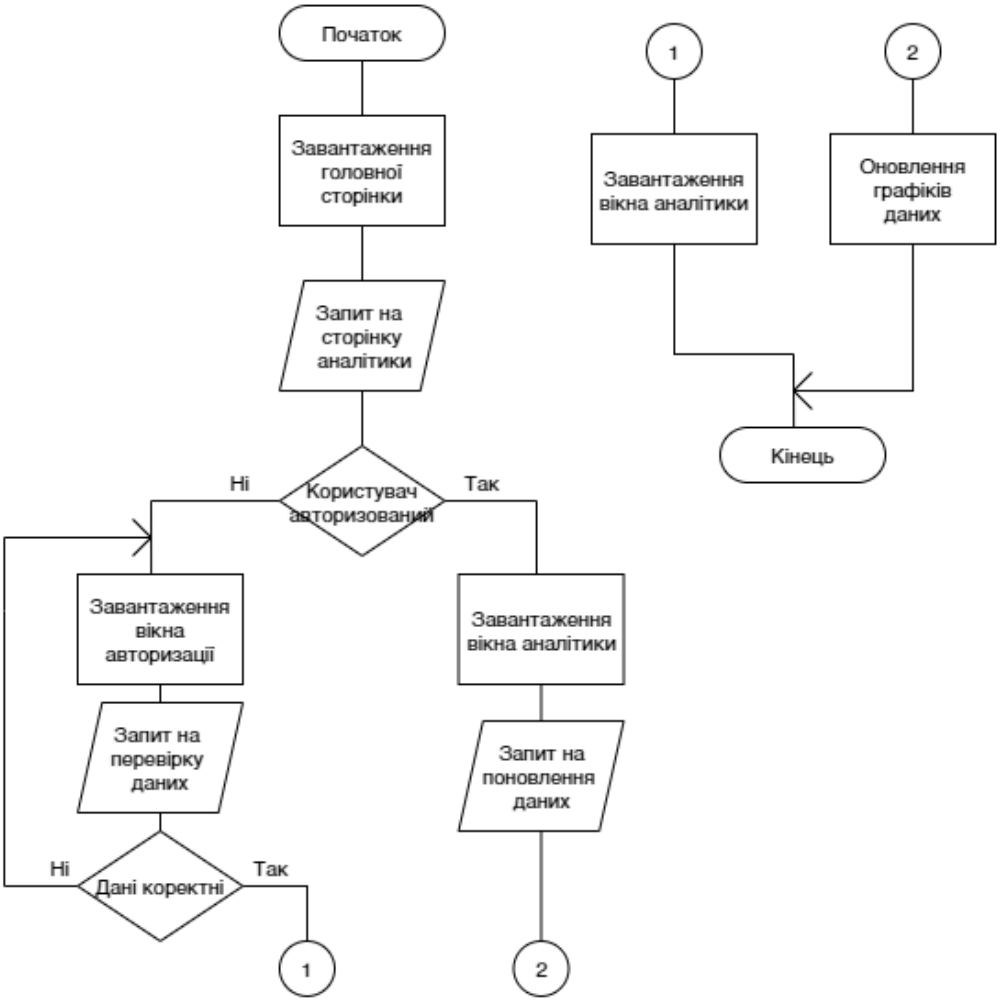
23. MongoDB Fundamentals: [Електронний ресурс]. Режим доступу до ресурсу: https://docs.mongodb.com/manual/faq/fundamentals/
24. Django Framework: [Електронний ресурс]. Режим доступу до ресурсу: https://developer.mozilla.org/ru/docs/Learn/Server-side/Django
25. Face-api.js — JavaScript API for Face Recognition in the Browser with tensorflow.js: [Електронний ресурс]. Режим доступу до ресурсу: https://itnext.io/face-api-js-javascript-api-for-face-recognition-in-the-browser-with-tensorflow-js-bcc2a6c4cf07
26. Tensorflow. Introduction, Architecture & Example: [Електронний ресурс]. Режим доступу до ресурсу: https://www.guru99.com/what-is-tensorflow.html
27. Keras: [Електронний ресурс]. Режим доступу до ресурсу: https://keras.io/guides/
28. OpenCV Library: [Електронний ресурс]. Режим доступу до ресурсу: https://opencv.org/
29. Мікросервісна архітектура: [Електронний ресурс]. Режим доступу до ресурсу: https://medium.com/@IvanZmerzlyi/microservices-architecture-461687045b3d

ДОДАТКИ

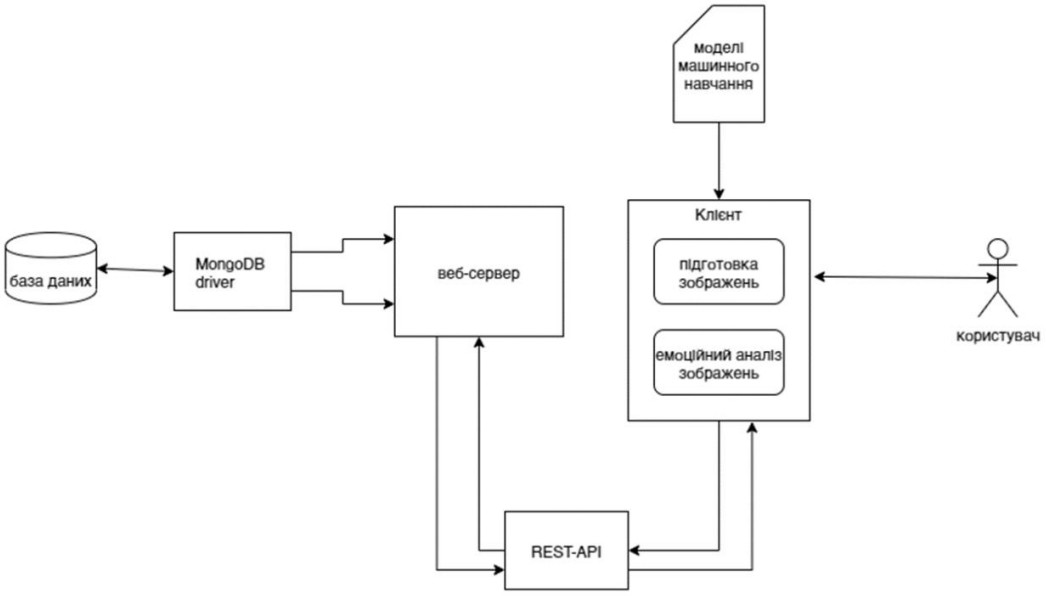
Додаток 1

Копії графічних матеріалів

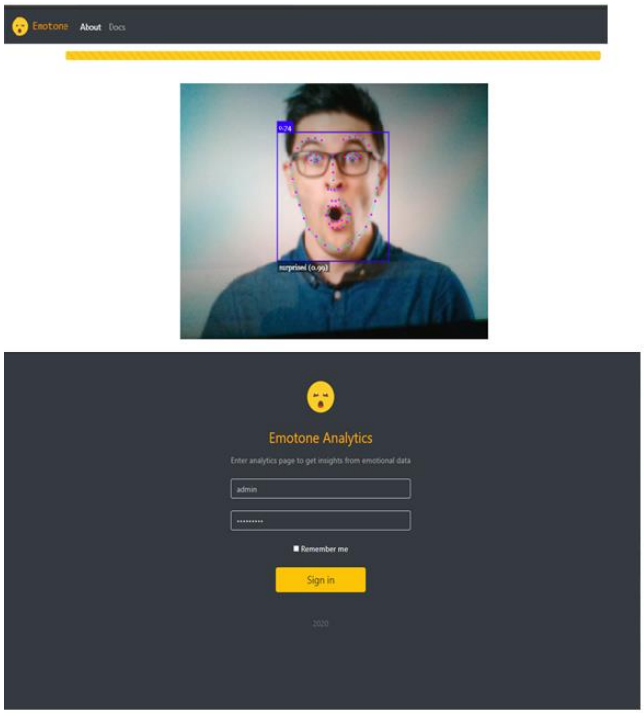
ДП.045440-06-99
Програмний застосунок для аналізу емоційного забарвлення зображень людини. Визначення емоційного забарвлення користувача. Схема алгоритму

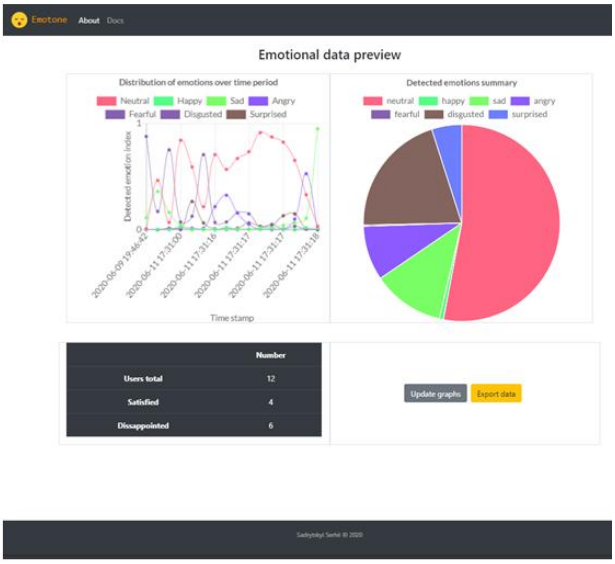


ДП.045440-07-99
Програмний застосунок для аналізу емоційного забарвлення зображень людини. Взаємодія графічних елементів. Схема алгоритму



Садрицький Сергій, група КП-61





Садрицький Сергій, група КП-61

Додаток 2

Лістинг програми

```
const express = require("express");
const bodyParser = require("body-parser");
const MongoClient = require("mongodb").MongoClient;
const path = require("path");
const ObjectID = require("mongodb").ObjectID;
const app = express();
const port = 8000;

app.set("view engine", "ejs");
app.use(express.json());
app.use(express.urlencoded({ extended: true }));

const viewsDir = path.join(__dirname, "views");
app.use(express.static(viewsDir));
app.use(express.static(path.join(__dirname, "./public")));
app.use(express.static(path.join(__dirname, "./weights")));
app.use(express.static(path.join(__dirname, "./dist")));

app.listen(port, () => {
  console.log("listen port 8000");
});

app.use(bodyParser.urlencoded({ extended: true }));

MongoClient.connect("mongodb://localhost:27017/data", {
  useUnifiedTopology: true,
}).then((client) => {
  const db = client.db("data");
  const quotesCollection = db.collection("expressions");

  app.get("/", (req, res) => res.sendFile(path.join(viewsDir,
"index.html")));
  app.get('/sendMeToTheAnalytics', (req, res) => res.redirect('/analytics'));

  app.post("/data", (req, res) => {
    const payload = JSON.parse(req.body.data);
    payload._id = new ObjectID();
    quotesCollection
```

```
      .insertOne(payload)
      // .then((result) => {
      //   console.log("saved");
      // })
      .catch((error) => console.error(error));
  });


  const numberOfSamples = 16; // number of time samples used for chart
generation
  app.get('/analytics', (req, res) => {
    db.collection('expressions').find({ "expressions": { $exists: true, $ne:
null } })
      .sort({ ts: -1 }).limit(numberOfSamples).toArray()
      .then(results => {


        const pieData = {
          neutral: 0.0,
          happy: 0.0,
          sad: 0.0,
          angry: 0.0,
          fearful: 0.0,
          disgusted: 0.0,
          surprised: 0.0
        }


        const lineData = {
          neutral: [],
          happy: [],
          sad: [],
          angry: [],
          fearful: [],
          disgusted: [],
          surprised: [],
          labels: []
        }
        const expressions = results.map(a => a.expressions)


        for (let i = 0; i < expressions.length; i++) {
```

```
          const expression = expressions[i]


          pieData.neutral = (parseFloat(pieData.neutral) +
parseFloat(expression.neutral)).toFixed(3)

          pieData.happy = (parseFloat(pieData.happy) +
parseFloat(expression.happy)).toFixed(3)

          pieData.sad = (parseFloat(pieData.sad) +
parseFloat(expression.sad)).toFixed(3)

          pieData.angry = (parseFloat(pieData.angry) +
parseFloat(expression.angry)).toFixed(3)

          pieData.fearful = (parseFloat(pieData.fearful) +
parseFloat(expression.fearful)).toFixed(3)

          pieData.disgusted = (parseFloat(pieData.disgusted) +
parseFloat(expression.disgusted)).toFixed(3)

          pieData.surprised = (parseFloat(pieData.surprised) +
parseFloat(expression.surprised)).toFixed(3)

        }


        // elemets go in DESCENDING order and we need ASCENDING order on the
line chart

        const reversed = results.reverse()

        for (let i = 0; i < reversed.length; i++) {


          let element = reversed[i];

          const expression = element.expressions


          lineData.neutral.push(parseFloat(expression.neutral).toFixed(3))

          lineData.happy.push(parseFloat(expression.happy).toFixed(3))

          lineData.sad.push(parseFloat(expression.sad).toFixed(3))

          lineData.angry.push(parseFloat(expression.angry).toFixed(3))

          lineData.fearful.push(parseFloat(expression.fearful).toFixed(3))


lineData.disgusted.push(parseFloat(expression.disgusted).toFixed(3))


lineData.surprised.push(parseFloat(expression.surprised).toFixed(3))


          const date = new Date(element.ts)

          lineData.labels.push(date.toISOString().replace(/T/, '
').replace(/\..+/, ''))

        }
```

```
        res.render('analitics.ejs', {
          pieData: pieData,
          lineData: lineData
        })
      })
      .catch(error => console.error(error))
  })
})
    let forwardTimes = [];
    let withBoxes = true;
    let withPoints = true;

    function stopVideo() {
      const videoEl = $("#inputVideo").get(0);
      videoEl.pause();
    }

    function onChangeHideBoundingBoxes(e) {
      withBoxes = !$(e.target).prop("checked");
    }

    // !
    function onChangeHideLandmarkPoints(e) {
      withPoints = !$(e.target).prop("checked");
    }

    // function updateTimeStats(timeInMs) {
    //   forwardTimes = [timeInMs].concat(forwardTimes).slice(0, 30);
    //   const avgTimeInMs =
    //     forwardTimes.reduce((total, t) => total + t) /
forwardTimes.length;
    //   $("#time").val(`${Math.round(avgTimeInMs)} ms`);
    //   $("#fps").val(`${faceapi.utils.round(1000 / avgTimeInMs)}`);
    // }
    const oneSecondTimeout = 50;
    async function onPlay() {
      const videoEl = $("#inputVideo").get(0);
      // console.log(videoEl);
```

```
    if (videoEl.paused || videoEl.ended || !isFaceDetectionModelLoaded())
      return setTimeout(() => onPlay());

    const options = getFaceDetectorOptions();
    const ts = Date.now();

    const result = await faceapi
      .detectSingleFace(videoEl, options)
      .withFaceLandmarks()
      .withFaceExpressions();
    // updateTimeStats(Date.now() - ts);

    if (result) {
      const canvas = $("#overlay").get(0);
      const dims = faceapi.matchDimensions(canvas, videoEl, true);
      // const displaySize = { width: videoEl.width, height: videoEl.height
};
      // const dims = faceapi.matchDimensions(canvas, displaySize);
      const resizedResult = faceapi.resizeResults(result, dims);
      const minConfidence = 0.05;
      if (withBoxes) {
        faceapi.draw.drawDetections(canvas, resizedResult);
      }
      if (withPoints) {
        faceapi.draw.drawFaceLandmarks(canvas, resizedResult);
      }
      faceapi.draw.drawFaceExpressions(canvas, resizedResult,
minConfidence);
    }
    await postResults(ts, result);
    setTimeout(() => onPlay(), oneSecondTimeout);
  }
  async function run() {
    // load face detection and face expression recognition models
    await changeFaceDetector(TINY_FACE_DETECTOR);
    await faceapi.loadFaceExpressionModel("/");
    await faceapi.loadFaceLandmarkModel("/");
```

```
      changeInputSize(224);
      // try to access users webcam and stream the images
      // to the video element
      const stream = await navigator.mediaDevices.getUserMedia({ video: {}
});
      const videoEl = $("#inputVideo").get(0);
      videoEl.srcObject = stream;
    }
    function updateResults() {}
    const userId = "default";
    function postResults(timestamp, expressions) {
      const payload = {};
      payload.user_id = userId;
      payload.ts = timestamp;
      payload.expressions = expressions?.expressions;

      $.post("http://localhost:8000/data", {
        type: "POST",
        dataType: "json",
        // async: false,
        data: JSON.stringify(payload),
      });
    }

    $(document).ready(function () {
      initFaceDetectionControls();
      run();
    });
const SSD_MOBILENETV1 = "ssd_mobilenetv1";
const TINY_FACE_DETECTOR = "tiny_face_detector";

let selectedFaceDetector = SSD_MOBILENETV1;

// ssd_mobilenetv1 options
let minConfidence = 0.5;

// tiny_face_detector options
let inputSize = 512;
```

```
let scoreThreshold = 0.5;

function getFaceDetectorOptions() {
  return selectedFaceDetector === SSD_MOBILENETV1
    ? new faceapi.SsdMobilenetv1Options({ minConfidence })
    : new faceapi.TinyFaceDetectorOptions({ inputSize, scoreThreshold });
}

function onIncreaseMinConfidence() {
  minConfidence = Math.min(faceapi.utils.round(minConfidence + 0.1), 1.0);
  $("#minConfidence").val(minConfidence);
  updateResults();
}

function onDecreaseMinConfidence() {
  minConfidence = Math.max(faceapi.utils.round(minConfidence - 0.1), 0.1);
  $("#minConfidence").val(minConfidence);
  updateResults();
}

function onInputSizeChanged(e) {
  changeInputSize(e.target.value);
  updateResults();
}

function changeInputSize(size) {
  inputSize = parseInt(size);

  const inputSizeSelect = $("#inputSize");
  inputSizeSelect.val(inputSize);
  //inputSizeSelect.material_select();
}

function onIncreaseScoreThreshold() {
  scoreThreshold = Math.min(faceapi.utils.round(scoreThreshold + 0.1), 1.0);
  $("#scoreThreshold").val(scoreThreshold);
  updateResults();
}
```

```
function onDecreaseScoreThreshold() {
  scoreThreshold = Math.max(faceapi.utils.round(scoreThreshold - 0.1), 0.1);
  $("#scoreThreshold").val(scoreThreshold);
  updateResults();
}

function onIncreaseMinFaceSize() {
  minFaceSize = Math.min(faceapi.utils.round(minFaceSize + 20), 300);
  $("#minFaceSize").val(minFaceSize);
}

function onDecreaseMinFaceSize() {
  minFaceSize = Math.max(faceapi.utils.round(minFaceSize - 20), 50);
  $("#minFaceSize").val(minFaceSize);
}

function getCurrentFaceDetectionNet() {
  if (selectedFaceDetector === SSD_MOBILENETV1) {
    return faceapi.nets.ssdMobilenetv1;
  }
  if (selectedFaceDetector === TINY_FACE_DETECTOR) {
    return faceapi.nets.tinyFaceDetector;
  }
}

function isFaceDetectionModelLoaded() {
  return !!getCurrentFaceDetectionNet().params;
}

async function changeFaceDetector(detector) {
  ["#ssd_mobilenetv1_controls", "#tiny_face_detector_controls"].forEach((id)
=>
    $(id).hide()
  );

  selectedFaceDetector = detector;
  const faceDetectorSelect = $("#selectFaceDetector");
```

```
  faceDetectorSelect.val(detector);
  //faceDetectorSelect.material_select();

  $("#loader").show();
  if (!isFaceDetectionModelLoaded()) {
    await getCurrentFaceDetectionNet().load("/");
  }

  $(`#${detector}_controls`).show();
  $("#loader").hide();
}

async function onSelectedFaceDetectorChanged(e) {
  selectedFaceDetector = e.target.value;

  await changeFaceDetector(e.target.value);
  updateResults();
}
async function requestExternalImage(imageUrl) {
  const res = await fetch('fetch_external_image', {
    method: 'post',
    headers: {
      'content-type': 'application/json'
    },
    body: JSON.stringify({ imageUrl })
  })
  if (!(res.status < 400)) {
    console.error(res.status + ' : ' + await res.text())
    throw new Error('failed to fetch image from url: ' + imageUrl)
  }

  let blob
  try {
    blob = await res.blob()
    return await faceapi.bufferToImage(blob)
  } catch (e) {
    console.error('received blob:', blob)
    console.error('error:', e)
```

```
    throw new Error('failed to load image from url: ' + imageUrl)
  }
}
function renderSelectList(selectListId, onChange, initialValue,
renderChildren) {
  const select = document.createElement('select')
  $(selectListId).get(0).appendChild(select)
  renderChildren(select)
  $(select).val(initialValue)
  $(select).on('change', (e) => onChange(e.target.value))
  $(select).material_select()
}


function renderOption(parent, text, value) {
  const option = document.createElement('option')
  option.innerHTML = text
  option.value = value
  parent.appendChild(option)
}
function initFaceDetectionControls() {
  const faceDetectorSelect = $("#selectFaceDetector");
  faceDetectorSelect.val(selectedFaceDetector);
  faceDetectorSelect.on("change", onSelectedFaceDetectorChanged);
 // faceDetectorSelect.material_select();
  // faceDetectorSelect.formSelect();


  const inputSizeSelect = $("#inputSize");
  inputSizeSelect.val(inputSize);
  inputSizeSelect.on("change", onInputSizeChanged);
  //inputSizeSelect.material_select();
  // inputSizeSelect.formSelect();
}
```

Додаток 3

Копія презентації

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

КАФЕДРА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ КОМП’ЮТЕРНИХ СИСТЕМ

ПРОГРАМНИЙ ЗАСТОСУНОК ДЛЯ АНАЛІЗУ ЕМОЦІЙНОГО ЗАБАРВЛЕННЯ ЗОБРАЖЕНЬ ЛЮДИНИ

Виконав: Садрицький Сергій

Керівник: доцент кафедри ПЗКС, к.т.н., доцент Сулема Є. С.

Київ – 2020

ПОСТАНОВКА ЗАДАЧІ

Мета проєкту: розробити програмний застосунок для виконання аналізу емоційного забарвлення зображень обличчя людини з використанням веб-камери персонального комп’ютера користувача.

ПОСТАНОВКА ЗАДАЧІ

1. Проаналізувати проблеми, які здатне вирішити програмне забезпечення для аналізу емоційного забарвлення зображень, та розглянути сучасні аналоги розроблюваної системи.
2. Розробити програмний застосунок для аналізу емоційного забарвлення зображень.
3. Проаналізувати якість розробленого програмного забезпечення та відповідність вимогам.

АКТУАЛЬНІСТЬ

1. Такі важливі сфери, як маркетинг, банкінг та медицина, потребують більш ефективних способів взаємодії з клієнтом.
2. Ринок систем емоційного аналізу невпинно зростає.
3. Активно збільшується актуальність систем на рубежі соціальних та технічних дисциплін.

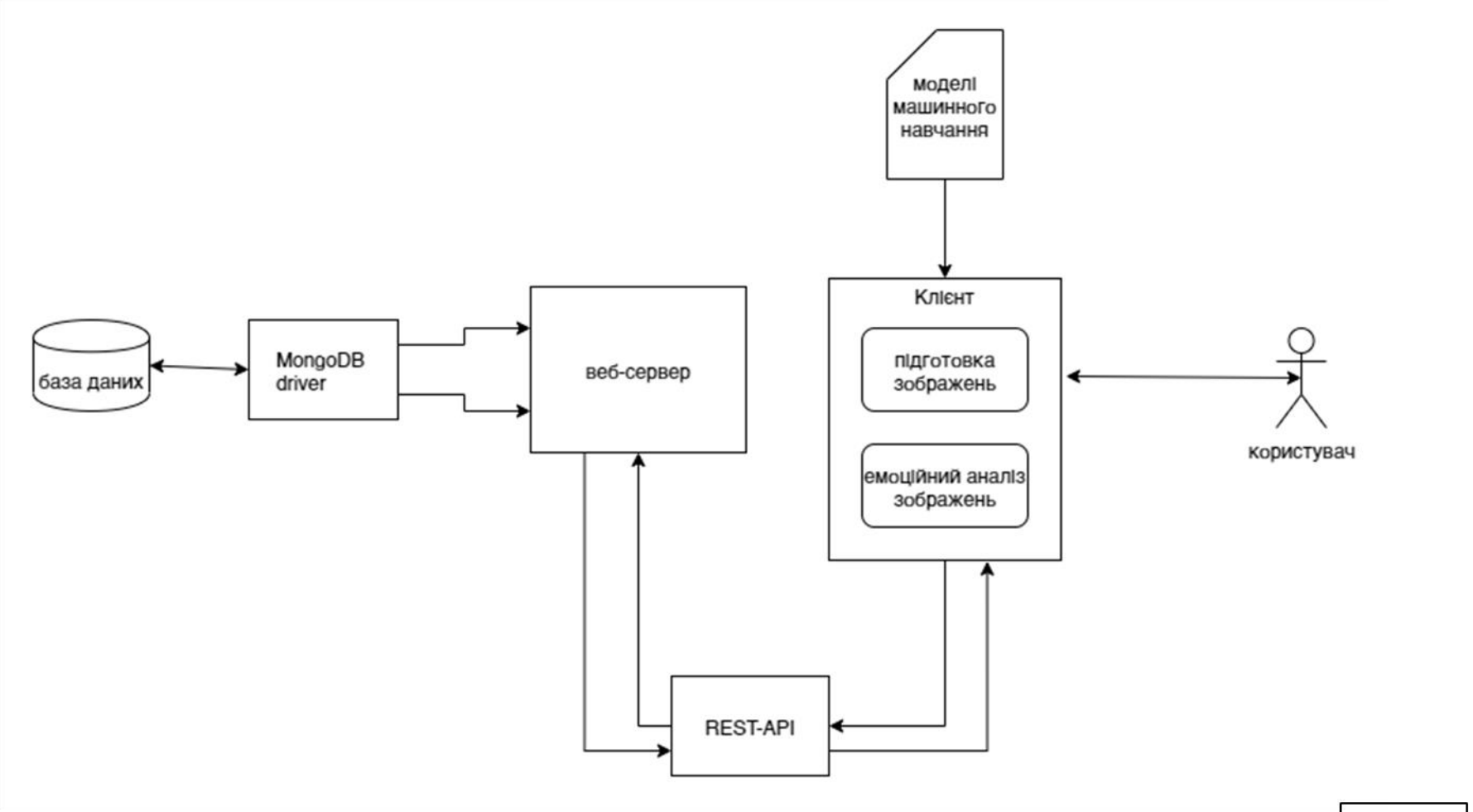
ІСНУЮЧІ РІШЕННЯ

EMO DETECT

:) Affectiva

nviso

АРХІТЕКТУРА СИСТЕМИ
Загальна структура застосунку



ЗАСОБИ РОЗРОБЛЕННЯ

- Мова програмування – *JavaScript:* крос-платформна об’єктно-орієнтована мультипарадигмальна мова розроблення веб-додатків.
- Серверна частина – *NodeJS:* відкрита платформа для створення високопродуктивних мережевих додатків, що базується на рушії V8, має асинхронну модель виконання та широкий вибір модулів.
- Клієнтська частина – *ReactJS:* відкрита JavaScript-бібліотека для створення клієнтських інтерфейсів, що характеризується гнучкістю побудови компонентів та роботи з об’єктною моделлю документа.

ЗАСОБИ РОЗРОБЛЕННЯ

Бібліотека для реалізації емоційного аналізу – face-api

Face-API – JavaScript-бібліотека, функціональність якої базується на відкритій бібліотеці машинного навчання TensorFlow, що, зокрема, призначена для вирішення завдань розпізнавання образів.

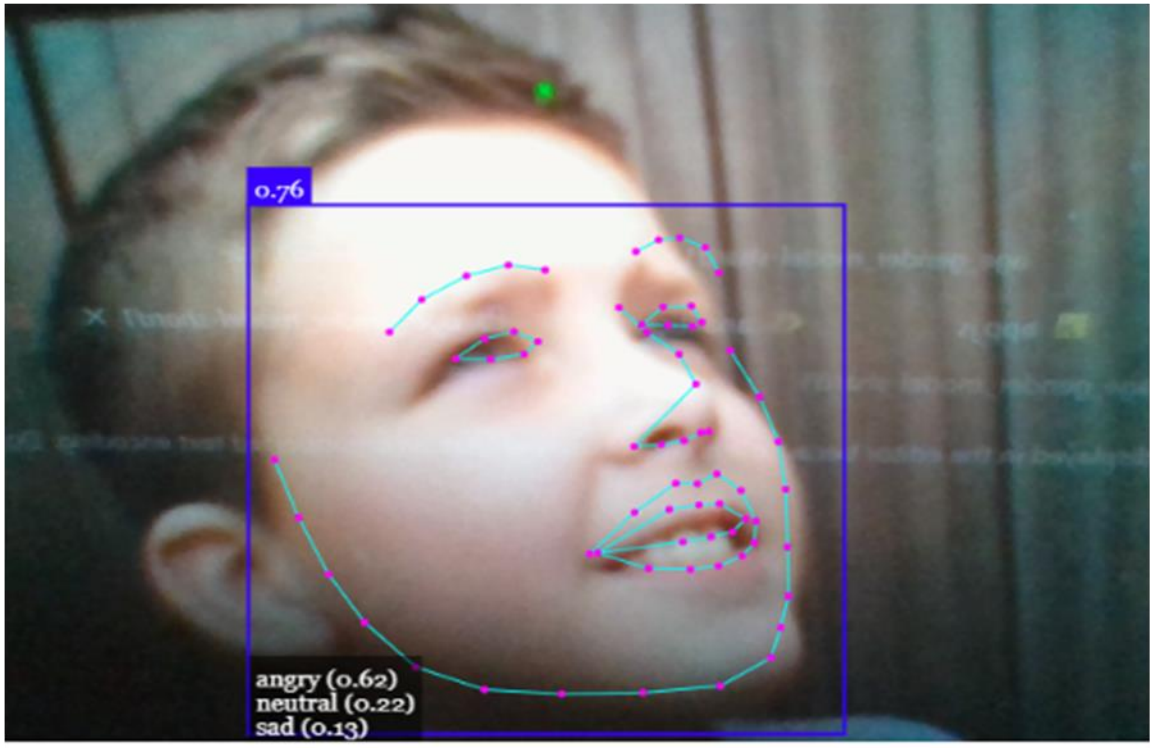
Face-API дозволяє проводити розпізнавання й аналіз зображень обличчя людини безпосередньо у браузері та на стороні сервера.

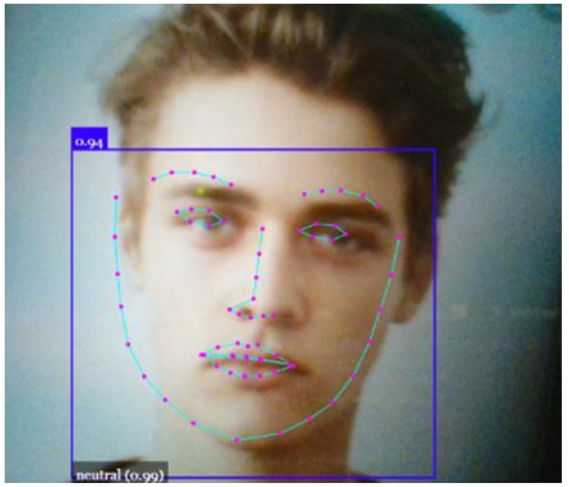
РОЗРОБЛЕНІ АЛГОРИТМИ ТА ЇХ РЕАЛІЗАЦІЯ

Розроблене ПЗ дозволяє визначати емоційний стан користувача на основі аналізу відеопотоку з веб-камери персонального пристрою, використовуючи алгоритми машинного навчання, що базуються на бібліотеці TensorFlow та попередньо підготовлених моделях. Для аналізу було обрано 7 типів емоцій – негативні, позитивні та нейтральний стан.

РОЗРОБЛЕНІ АЛГОРИТМИ ТА ЇХ РЕАЛІЗАЦІЯ

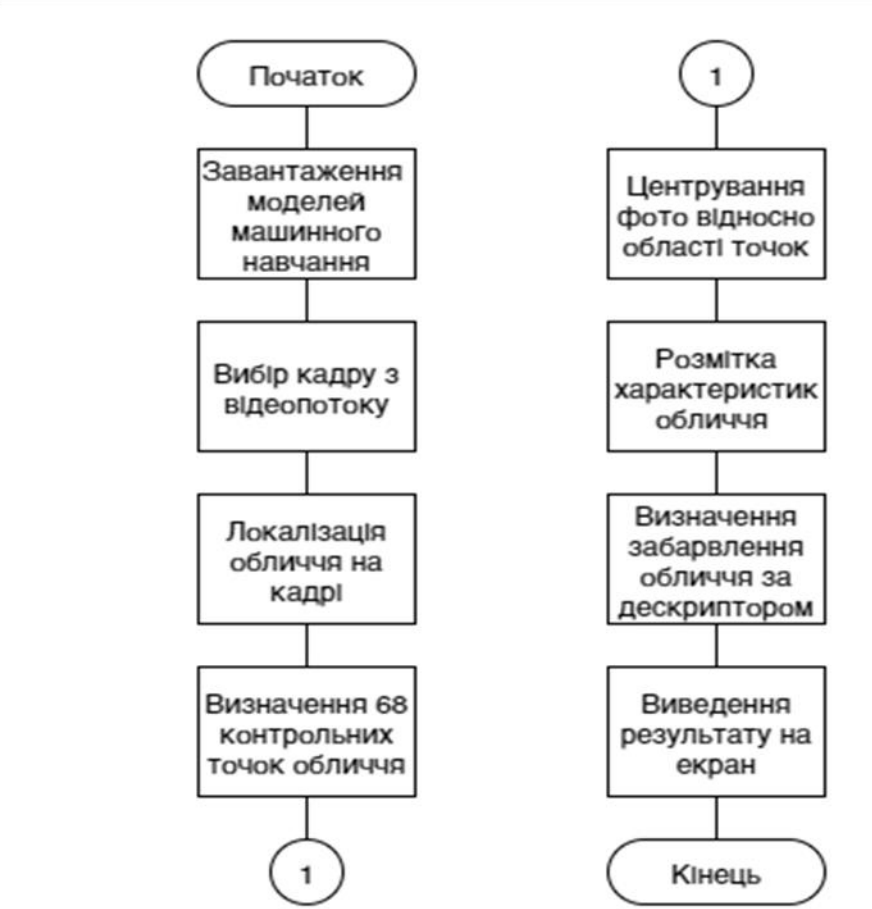
Алгоритми дозволяють проаналізувати відеопотік, побудувати сітку контрольних точок і за ними визначити характеристику емоції.





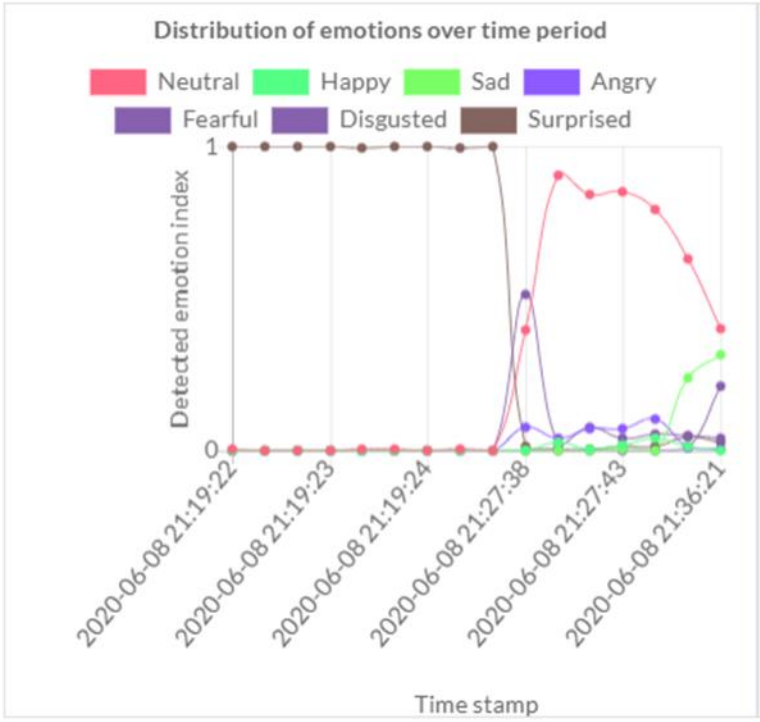
РОЗРОБЛЕНІ АЛГОРИТМИ ТА ЇХ РЕАЛІЗАЦІЯ

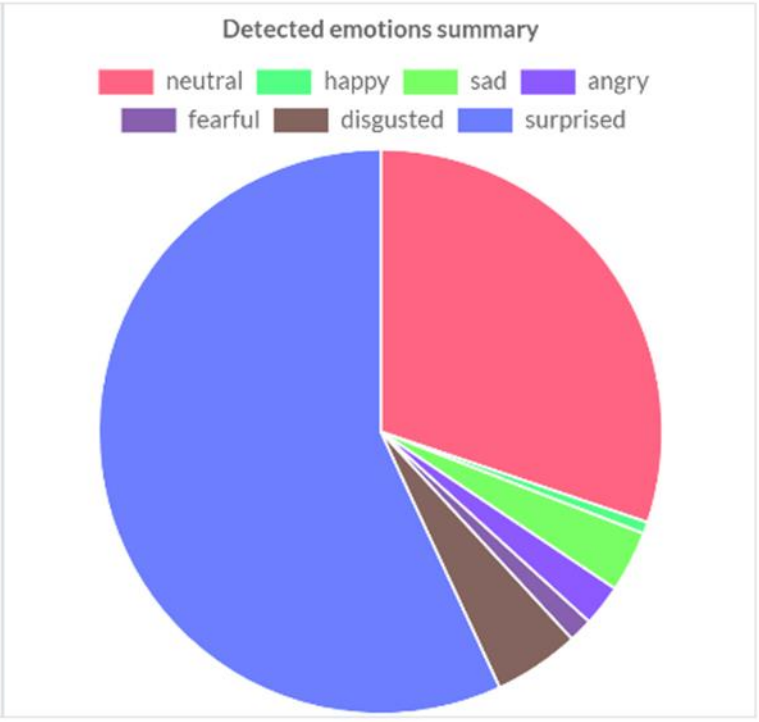
Алгоритм визначення емоційного забарвлення



РОЗРОБЛЕНІ АЛГОРИТМИ ТА ЇХ РЕАЛІЗАЦІЯ

Дані емоційного забарвлення оформлюються у вигляді графіків та діаграм



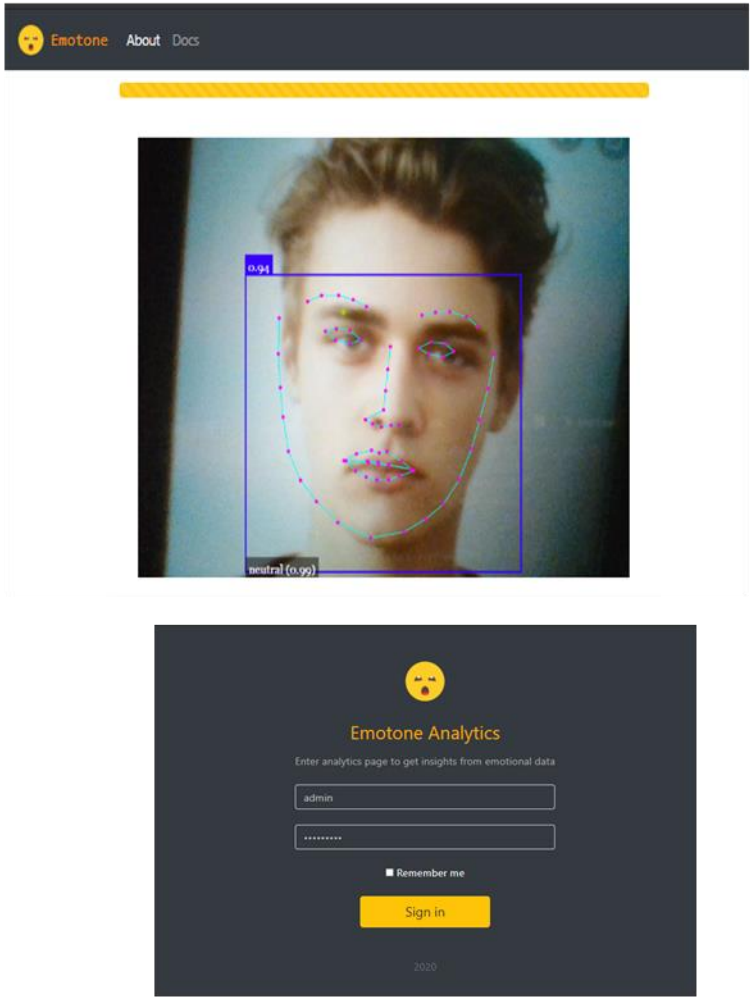


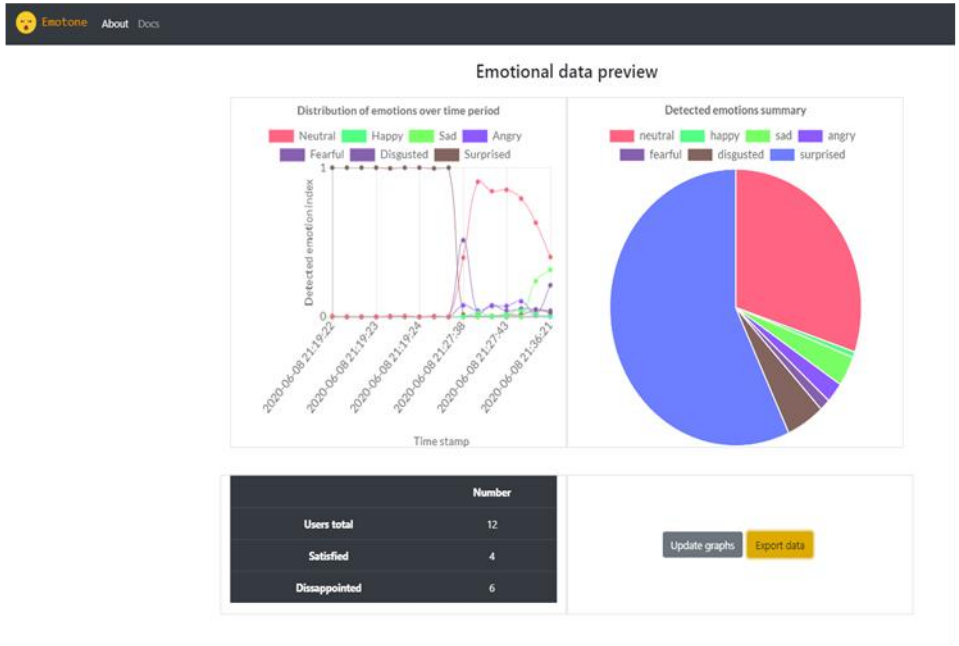
КРИТЕРІЇ ОЦІНЮВАННЯ ЯКОСТІ РОЗРОБЛЕНОГО ПЗ

Відповідність висунутим вимогам до програмного застосунку:

- забезпечення обробки й аналізу зображень в реальному часі,
- перегляд даних про емоційне забарвлення обличчя користувача,
- подання отриманих даних у графічному вигляді,
- зручний користувацький інтерфейс

РОЗРОБЛЕНА СИСТЕМА





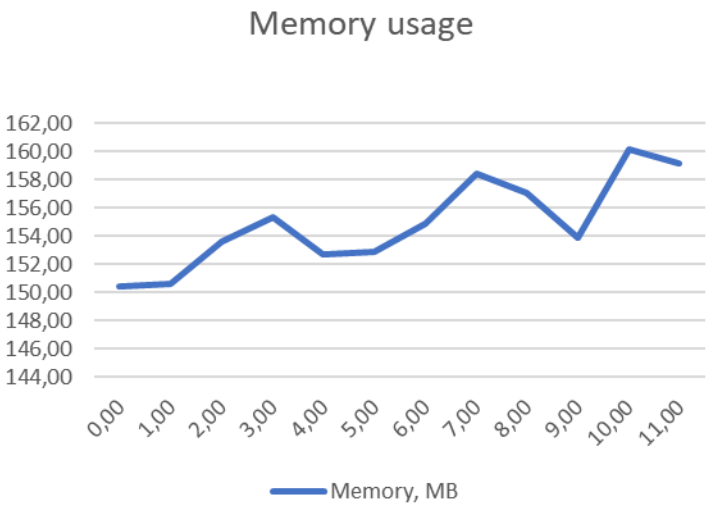
РЕЗУЛЬТАТИ ТЕСТУВАННЯ

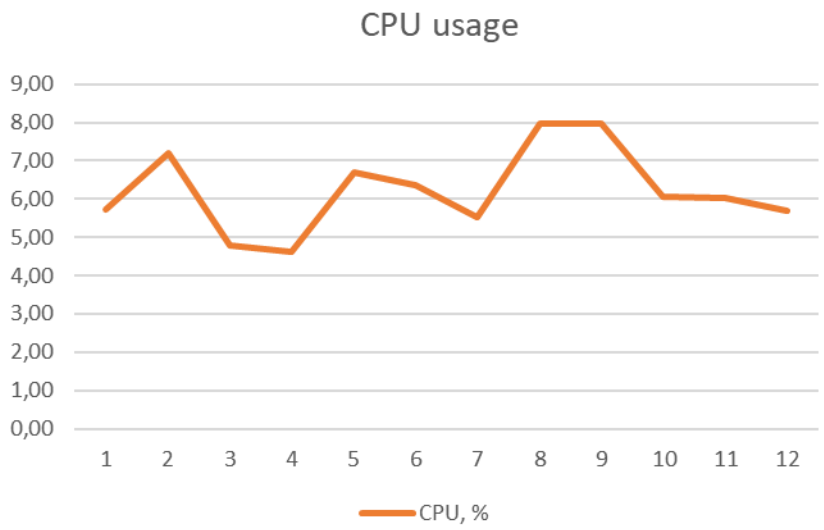
Виконане тестування обміну даними серверної частини з БД показало, що дані відображаються коректно.

При тестуванні інтерфейсу користувача виявлено, що елементи вікон дозволяють зручно маніпулювати функціями застосунку.

Тестування модуля оброблення зображень продемонструвало його швидкодію та відносно низьку ресурсовитратність при роботі в браузері.

РЕЗУЛЬТАТИ ТЕСТУВАННЯ





Конфігурація для тестування:
CPU: Intel Core i5-6200 dual-core (2.3 GHz)
GPU: Intel hd 520
Memory: 16 GB
ОС: Windows 10 Pro 64bit
Браузер: Google Chrome

ВИСНОВКИ

1. Проаналізовано існуючі програмні рішення та виявлено їх особливості.
2. Визначено архітектуру програмного застосунку, основні компоненти та засоби для їх реалізації.
3. Визначено функціональні вимоги до ПЗ, зокрема вимоги до серверної та клієнтської частин, інтерфейсу та модуля оброблення зображень.
4. Визначено критерії аналізу емоційного забарвлення зображень.
5. Розроблено програмний застосунок, що в повному обсязі відповідає функціональним вимогам.
6. Тестування програмного застосунку виконано в повному обсязі згідно наявної методики тестування.

Дякую за увагу!

Факультет прикладної математики

Кафедра програмного забезпечення комп'ютерних систем

«ЗАТВЕРДЖЕНО»

Науковий керівник кафедри

_____ Іван ДИЧКА

«___» _____ 2019 р.

ПРОГРАМНИЙ ЗАСТОСУНОК ДЛЯ АНАЛІЗУ ЕМОЦІЙНОГО ЗАБАРВЛЕННЯ ЗОБРАЖЕНЬ ЛЮДИНИ

Програма та методика тестування

ДП.045440-04-51

«ПОГОДЖЕНО»

Керівник проєкту:

_____ Євгенія СУЛЕМА

Нормоконтроль:

_____ Микола ОНАЙ

Виконавець:

_____ Сергій САДРИЦЬКИЙ

2019

ЗМІСТ

1. ОБ’ЄКТ ВИПРОБУВАНЬ

Програмний застосунок для аналізу емоційного забарвлення зображень обличчя людини, розроблений з використанням мови програмування JavaScript, фреймворків NodeJS для серверної частини та ReactJS для клієнтської частини.

2. МЕТА ТЕСТУВАННЯ

У процесі тестування має бути перевірено наступне:

1) відповідність реалізованої функціональності до заявленої у вимогах;
2) коректність і точність роботи алгоритму розпізнавання емоційного забарвлення зображень;
3) коректність відображення даних у базі даних;
4) забезпечення належного рівня швидкодії застосунку;
5) зручність роботи з застосунком.

3. МЕТОДИ ТЕСТУВАННЯ

При тестуванні даного програмного застосунку був використаний метод Gray Box. При такому підході перевіряється як якість написаного коду, так і безпосередньо програмний продукт при взаємодії з користувачем на відповідність функціональним вимогам. Тестування проведене на рівні «системного тестування».

Використовуються наступні методи:

1) аналіз критичних значень для перевірки коректності розпізнавання емоційного забарвлення зображень;
2) тестування швидкодії та стабільності роботи програмного застосунку у браузері;
3) тестування інтерфейсу.

5. ЗАСОБИ ТА ПОРЯДОК ТЕСТУВАННЯ

Тестування програмного застосунку виконується з використанням таких методів:

1) тестування застосунку при різних умовах освітлення;
2) тестування при віддаленні на різну відстань від камери пристрою;
3) тестування працездатності додатку за наявності перешкод перед користувачем;
4) тестування стабільності роботи застосунку на малопотужних пристроях;
5) тестування коректності подання даних в базі даних;
6) тестування коректності подання даних у графічному вигляді;
7) тестування інтерфейсу на відповідність критеріям зручності.

Факультет прикладної математики

Кафедра програмного забезпечення комп'ютерних систем

«ЗАТВЕРДЖЕНО»

Науковий керівник кафедри

_____ Іван ДИЧКА

«___» _____ 2020 р.

ПРОГРАМНИЙ ЗАСТОСУНОК ДЛЯ АНАЛІЗУ ЕМОЦІЙНОГО ЗАБАРВЛЕННЯ ЗОБРАЖЕНЬ ЛЮДИНИ

Керівництво користувача

ДП.045440-05-34

«ПОГОДЖЕНО»

Керівник проєкту:

_____ Євгенія СУЛЕМА

Нормоконтроль:

_____ Микола ОНАЙ

Виконавець:

_____ Сергій САДРИЦЬКИЙ

2020

ЗМІСТ

1. Опис структури програмного застосунку

Програмний застосунок для аналізу емоційного забарвлення зображень людини складається зі статичних та веб-сторінок, наповнення яких генерується сервером динамічно. Статичними є головна сторінка застосунку та сторінка авторизації, а динамічно генерованою є сторінка перегляду статистики. Користувацький інтерфейс додатку виконано англійською мовою.

Кожна сторінка містить елементи з посиланням, за допомогою яких користувач зможе перейти на головну сторінку додатку та елементи з посиланнями на інші сторінки.

2. Опис вмісту статичних web-сторінок

Початкова сторінка застосунку

Головна сторінка додатку містить основні компоненти інтерфейсу, що слугують для виведення та керування відео потоком. Також на головному вікні можна знайти посилання на додаткову інформацію про додаток у верхній панелі.

Вигляд головного вікна проілюстровано на рис. 1.

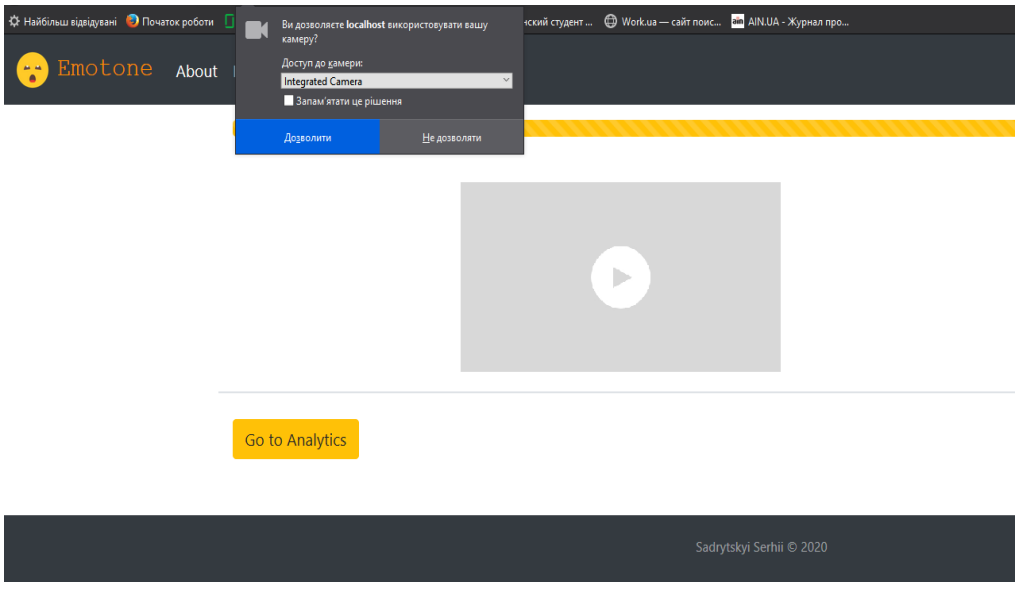


Рис. 1. Вигляд початкової сторінки додатка

Одразу при вході користувача у додаток з’являється вікно з запитом на доступ до веб-камери пристрою. Після того, як користувач дає дозвіл, у вікно трансляції виводиться відео потік та система починає аналізувати його вміст. Для наочності користувач може ввімкнути відображення контрольних точок, які визначають положення рис обличчя та рамку, що обмежує область контрольних точок.

Результат процесу побудови контрольних точок проілюстровано на рис. 2.

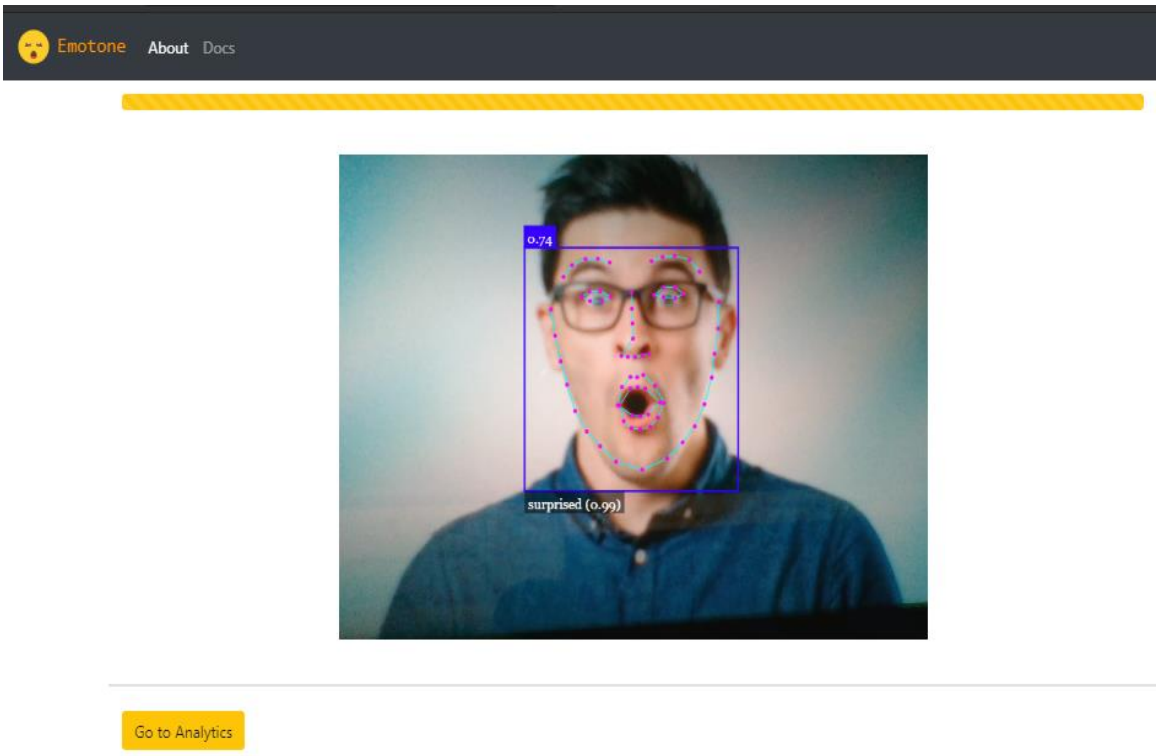


Рис. 2. Вигляд головного вікна

Сторінка авторизації

Дане вікно містить поля для вводу персональних даних адміністратора для перегляду графічних аналітичних даних про користувачів.

Вигляд вікна авторизації наведено на рис. 3.

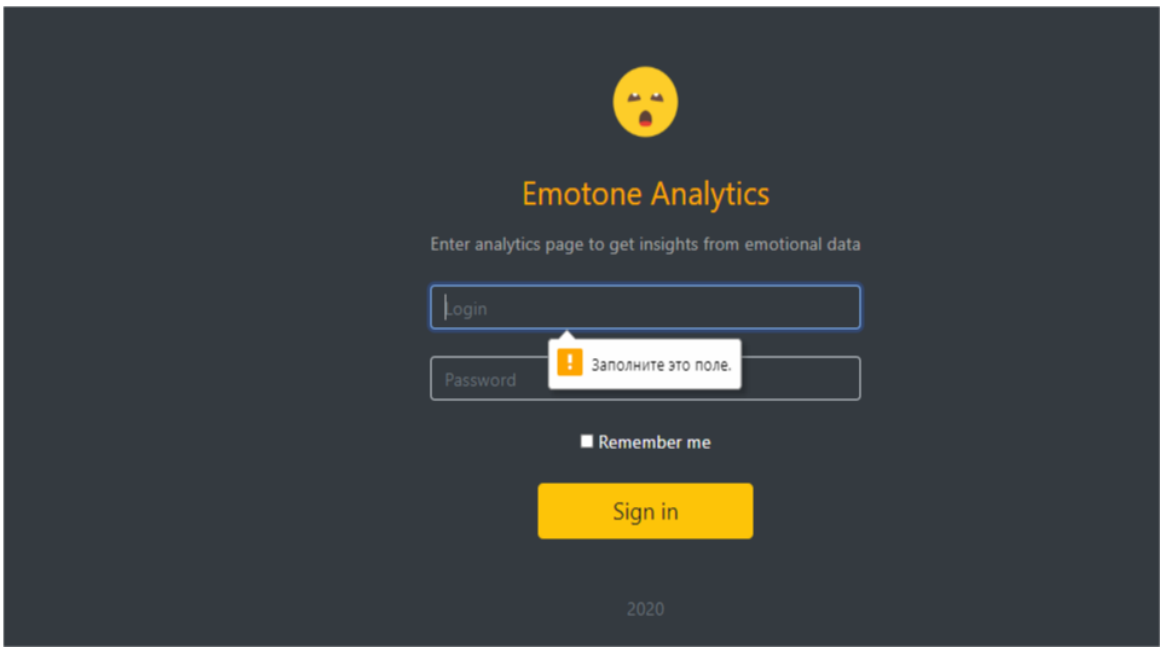


Рис. 3. Вигляд сторінки авторизації

3. Процедура авторизації користувача

Для виконання авторизації користувач має ввести мвої логін та пароль та натиснути кнопку «Sign In», після чого відбувається перехід на сторінку аналітики.

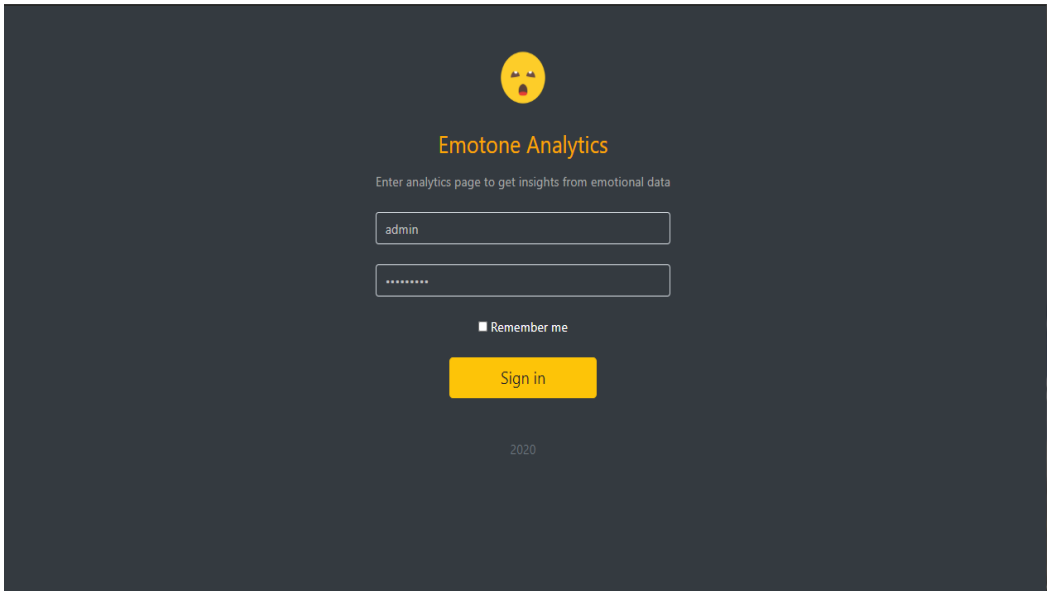


Рис. 4. Сторінка авторизації

4. Процедура отримання статистики користувачів

Після авторизації користувач переходить на сторінку статистики. На ній містяться графіки та діаграми, що відображають поточний стан емоцій користувачів. Наприклад, на графіку зліва (рис. 5), зображено розподіл появи емоцій за певний період часу, а на круговій діаграмі справа – співвідношення між різними видами емоцій за весь період часу.

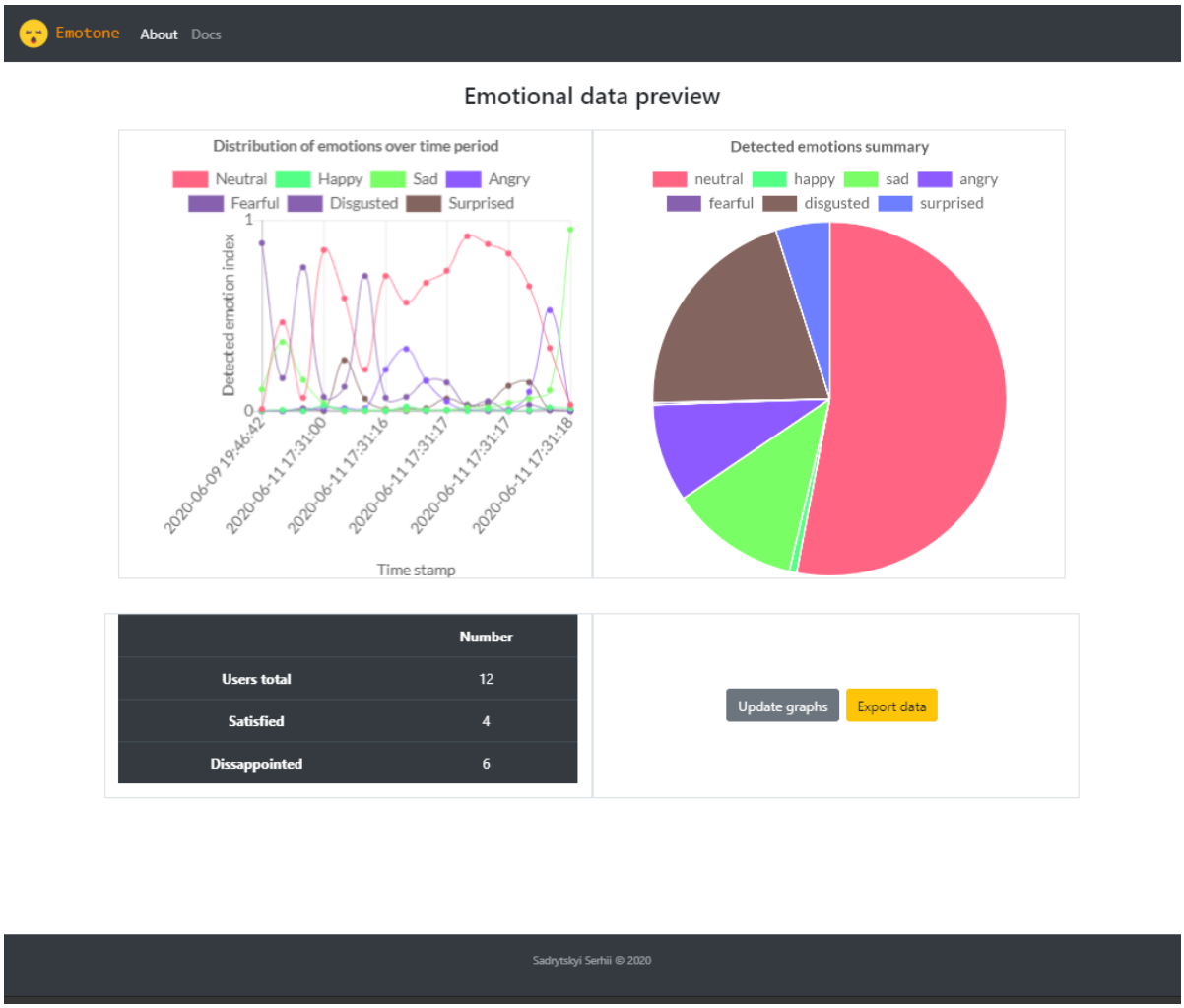


Рис. 5. Сторінка аналітики